MODÈLES ET ALGORITHMES POUR LA SEGMENTATION DE SÉQUENCES BIOLOGIQUES ET LA RECONSTRUCTION DE LEURS HISTOIRES ÉVOLUTIVES

par

Esaie Kuitche Kamela

Thèse présentée au Département d'informatique
en vue de l'obtention du grade de philosophiæ doctor (Ph.D.)

FACULTÉ DES SCIENCES

UNIVERSITÉ DE SHERBROOKE

Sherbrooke, Québec, Canada, 22 juillet 2020

Le 22 juillet 2020

Le jury a accepté la thèse de Esaie Kuitche Kamela dans sa version finale

Membres du jury

Professeure Aïda Ouangraoua
Directrice
Département Informatique

Professeur Shengrui Wang
Codirecteur
Département Informatique

Professeur Pierre-Étienne Jacques
Membre interne
Département Biologie

Professeur
Mathieu Blanchette
Membre externe
Département Computer Science
McGill University, QC, CA

Professeur Michael Blondin
Président-rapporteur
Département Informatique

Sommaire

L'informatique est de plus en plus utilisée pour résoudre des problèmes dans divers domaines. C'est ainsi qu'avec l'accroissement des données biologiques générées par les techniques expérimentales à haut débit, la bio-informatique intervient pour tirer profit de ces masses de données et contribuer à l'avancement des connaissances en sciences biologiques. La bio-informatique est un domaine interdisciplinaire ayant pour but d'étudier et de résoudre des problèmes computationnels issus des sciences biologiques.

Un des problèmes intemporels étudié en bio-informatique est la reconstruction de l'histoire évolutive de génomes, qui sous-entend essentiellement celle des gènes. Les gènes sont le support de l'information génétique et sont les unités de base de l'hérédité. De nos jours, un grand nombre de maladies, telles les cancers, ont une base génétique. Une bonne compréhension de l'évolution des gènes permettrait de mieux comprendre les processus impliqués dans ces maladies pour mieux les traiter. De plus, les connaissances sur l'évolution de gènes sont utiles pour la prédiction et l'annotation de nouveaux gènes.

Il a été montré que les gènes eucaryotes subissent un phénomène d'épissage alternatif qui permet aux gènes de produire plusieurs transcrits différents afin de se diversifier fonctionnellement. C'est dans ce contexte que se situe cette thèse de doctorat. L'objectif de la thèse est de définir des modèles et des algorithmes efficaces et précis pour la segmentation de séquences biologiques et la reconstruction de leurs histoires évolutives en tenant compte de l'épissage alternatif. Dans cette thèse, j'ai contribué à accroître les connaissances scientifiques en introduisant et en formalisant des modèles d'évolution de transcrits et de gènes. Nous avons proposé deux algorithmes pour la segmentation de transcrits alternatifs. Nous avons également proposé un outil de simulation de l'évolution des séquences biologiques et un outil de visuali-

sation de coévolution. Pour chacun des modèles et algorithmes proposés, nous avons développé des applications pour permettre l'utilisation facile de nos outils.

Mots-clés: Épissage alternatif ; Arbres phylogénétiques ; Segmentation de séquences, Simulation de l'évolution de séquences ; Réconciliation phylogénétiques ; Arbres de transcrits ; Arbres de gènes.

Dédicace

À ma tendre épouse Floriane Kuitche.
À mes jumeaux Daisy Grace et Samuel Legrand.

Remerciements

Je remercie tout d'abord chaudement ma directrice de recherche Pre Aïda Ouangraoua de m'avoir initié à l'univers de la recherche. Tout au long de mes années de recherches auprès d'elle dans son laboratoire, j'ai pu bénéficier de son attention, de son orientation et de sa rigueur scientifique. Aïda m'a donné la latitude d'explorer plusieurs axes de recherche ; ce qui a contribué à accroître mon éveil scientifique et mon autonomie. L'ambiance conviviale du laboratoire CoBIUS qu'elle dirige a été pour moi un cadre paisible dans lequel j'ai effectué et réalisé mes recherches.

Je tiens également à remercier mon codirecteur Pr Shengrui Wang pour son soutien tout au long de ma thèse. Ses conseils, ses questions et son appui ont été précieux pour moi.

Je remercie grandement Pr Benoît Chabot pour ses remarques et ses suggestions ; elles m'ont permis de comprendre davantage certaines problématiques biologiques.

Je remercie les membres du jury d'avoir accepté l'évaluation de ma thèse et de m'avoir fait part de suggestions et de commentaires qui contribuent à améliorer cette thèse. Je remercie aussi tous les professeurs du Département d'informatique, de la Faculté des sciences ainsi que les techniciens et tout le personnel administratif pour l'aide qu'ils m'ont apportée durant ma thèse.

Je remercie tous ceux avec qui j'ai pu collaborer à un moment donné dans ma thèse. Je citerai particulièrement Pr Manuel Lafond et Pr Christophe Dessimoz.

Je remercie tous mes collègues du laboratoire CoBiUS avec lesquels j'ai partagé de très beaux moments de travail dans une ambiance fraternelle. Par ordre d'arrivé, je cite Safa Jammali, Sarah Belhamiti, Jean-David Aguilar, Michaël Luce, Jean-Pierre Glouzon, Nilson Da Rocha Coimbra, Anaïs Vannutelli, Ali Fotouhi, Marc-André Bossanyi, Yoann Anselmetti, Ibrahim Chegrane, Abigail Djossou et Davy Ouédraogo.

Remerciements

Je remercie tout particulièrement mes professeurs du Département du génie informatique de l'École Nationale Supérieure Polytechnique de Yaoundé pour leur soutien. Je cite en particulier Pr Thomas Bouetou qui m'a recommandé auprès de Pre Ouangraoua pour cette thèse, Pr Georges Édouard Kouamou et Pre Claude Marie Ngabireng. Je tiens à remercier toute ma famille qui m'a soutenu tout au long de ma thèse et qui a dû subir mon indisponibilité : je vous en serai à jamais reconnaissant.

Un merci tout particulier à ma tendre épouse Floriane Kuitche de m'avoir soutenu et encouragé durant toutes ces années. Sache qu'il y a un peu de toi dans chacune de mes réalisations liées à cette thèse.

Abréviations

A : Adénine

ADN : Acide désoxyribonucléique

ARN : Acide ribonucléique

CDS : *Coding Sequence*

C : Cytosine

DI : Département d'informatique

G : Guanine

T : Thymine

U : Uracile

UdeS : Université de Sherbrooke

UPGMA : *Unweighted Pair Group Method Arithmetic Mean*

Table des matières

Liste des figures

Liste des tableaux

Introduction

Contexte

La comparaison des génomes permet de mieux comprendre le fonctionnement des organismes, des espèces et leur évolution. L'annotation des génomes apporte des connaissances supplémentaires sur les processus biologiques qui se déroulent dans les cellules du vivant. Le génome écrit sur un alphabet d'Acide DésoxyriboNucléique (ADN) qui est souvent décrit comme le livre de la vie est unique à chaque être vivant. Il contient toutes les informations régissant le développement, le fonctionnement et la reproduction des êtres vivants. Depuis plus d'un siècle, de nombreuses recherches ont été menées afin de décrypter chacune des pages de ce précieux livre. Une meilleure compréhension du fonctionnement des séquences d'ADN permet de faire face aux facteurs de dysfonctionnement de l'ADN des êtres vivants. Les recherches sur l'ADN sont aussi diverses qu'intéressantes. Des exemples de questions étudiées en bio-informatique sont l'annotation des génomes, la comparaison des séquences biologiques, la segmentation des séquences biologiques, la prédiction des fonctions des séquences et la reconstruction de phylogénies. Dans le cadre de la reconstruction des phylogénies, il peut s'agir de reconstruire l'évolution d'un ensemble d'espèces, d'un ensemble de gènes ou de tout autre ensemble d'entités reliées en étudiant leurs séquences. Les travaux sur la segmentation des séquences servent par exemple à identifier et caractériser des groupes de séquences afin de déduire les fonctions biologiques de nouvelles séquences. Les travaux sur la comparaison des génomes de l'humain et de la souris ont permis par exemple de constater qu'environ 99% des gènes humains ont des homologues chez la souris. Cette information permet de comprendre que les humains et les souris partagent une grande proportion de fonctions biologiques com-

parables. En plus de contribuer à accroître notre compréhension du fonctionnement des espèces, ces connaissances sont utilisées pour la recherche de traitements pour des maladies en laboratoire. D'autres sujets d'étude s'intéressent également à reconstruire l'histoire évolutive des gènes. Ils ont pour but de découvrir comment un ensemble de gènes a évolué depuis un ancêtre commun. De nombreuses études ont montré que les gènes eucaryotes ont la capacité de produire plus d'un transcrit par le mécanisme de l'épissage alternatif, et que ces transcrits peuvent avoir des fonctionnalités différentes, permettant ainsi aux gènes de diversifier leurs fonctions. De ce fait, la reconstruction des phylogénies de gènes et de transcrits doit prendre en compte le mécanisme d'épissage alternatif, qui joue un rôle très important du point de vue fonctionnel. C'est dans ce contexte que s'inscrit l'objectif de cette thèse : la segmentation des séquences et la reconstruction de leurs histoires évolutives en tenant compte de l'épissage alternatif.

Objectifs

Étant donné une famille de gènes qui est composée d'un ensemble de gènes homologues appartenant à un ensemble d'espèces, et un ensemble de transcrits produits par les gènes de cette famille, l'objectif général de cette thèse est de proposer une méthode de segmentation et de reconstruction de l'histoire évolutive des séquences issues de cette famille de gènes. Plus spécifiquement, les questions étudiées sont les suivantes :

1. Définir un modèle pour l'évolution des transcrits et pour l'évolution des gènes tenant compte de l'épissage alternatif ;
2. Définir une méthode pour la reconstruction des arbres de gènes et des arbres de transcrits tenant compte de l'épissage alternatif ;
3. Définir un modèle pour la segmentation des séquences de transcrits tenant compte de l'épissage alternatif ;
4. Développer un outil de visualisation spécifique à la coévolution d'espèces, de gènes et de transcrits.

Méthodologie

Dans le cadre de cette thèse, j'ai suivi un processus exploratoire qui consiste à étudier un ensemble de problématiques connexes. La première problématique abordée est celle de la reconstruction des phylogénies des transcrits et des gènes. Après avoir cerné les limites des travaux actuels, nous avançons qu'il est important de distinguer les phylogénies de gènes de celles des transcrits. Par la suite, nous présentons un modèle de coévolution entre des gènes et des transcrits, qui mène à deux problèmes algorithmiques portant sur la reconstruction d'arbres de gènes et de transcrits. Un algorithme basé sur la segmentation hiérarchique avec chevauchement est également proposé. Ensuite, nous nous intéressons à l'évaluation de la qualité des arbres phylogénétiques obtenus. Étant donnée l'absence de données de référence réelles sur l'évolution d'ensembles de transcrits, nous explorons l'alternative de la simulation de l'évolution des gènes et transcrits. C'est ainsi qu'un modèle et une application pour simuler l'évolution des gènes et des transcrits en tenant compte de l'épissage alternatif ont été proposés. Ensuite, nous avons revisité l'algorithme de segmentation hiérarchique de séquences biologiques afin de repousser un certain nombre de limites. Le nouvel algorithme proposé intègre une mesure de similarité qui, en plus de tenir compte de la séquence, prend également en compte la structure d'épissage. Cet algorithme présente aussi une meilleure approche pour identifier le nombre optimal de groupes et la répartition des séquences dans ces groupes. L'utilisation de cet algorithme permet d'améliorer la qualité des arbres reconstruits. Enfin se pose la question de la visualisation. Nous avons développé le premier outil de visualisation conjointe de l'évolution d'un ensemble de transcrits dans un arbre de gènes et de l'évolution d'un ensemble de gènes dans un arbre d'espèces. Les problématiques étudiées dans cette thèse s'intègrent dans le thème global de recherche du laboratoire de recherche CoBiUS qui s'intéresse au développement de modèles et algorithmes innovants pour la reconstruction de l'évolution des génomes et de leurs composants.

Contributions

Nos travaux contribuent à l'avancement de la science de deux manières : par les publications scientifiques et par les applications développées. Nous abordons la question de reconstruction conjointe d'arbres de transcrits et d'arbres de gènes dans notre premier article intitulé *«Reconstructing protein and gene phylogenies using reconciliation and soft-clustering»* et publiée dans Journal of Bioinformatics and Computational Biology. Nous y proposons une méthode de segmentation des séquences biologiques et une autre de correction des arbres de gènes. Notre second article intitulé *«SimSpliceEvol : Alternative splicing-aware simulation of biological sequence evolution»* et publié dans BMC Bioinformatics décrit formellement le modèle d'évolution d'un ensemble de transcrits dans un arbre de gènes, et permet de simuler l'évolution des séquences de transcrits et de gènes en tenant compte de l'épissage alternatif. Les séquences obtenues peuvent ensuite servir de références pour des tests de méthodes de reconstruction phylogénétique. Dans notre troisième article intitulé *«Choosing representative proteins based on splicing structure similarity improves the accuracy of gene tree reconstruction»* et soumis à BioRXiv, nous proposons une nouvelle mesure de similarité adaptée aux transcrits alternatifs, et nous présentons un nouvel algorithme pour la segmentation non supervisée des transcrits et la reconstruction d'arbres de gènes. Dans notre quatrième article intitulé *«DoubleRecViz : A Web-Based Tool for Visualizing Transcript-Gene-Species reconciliation»* et soumis à Bioinformatics, nous proposons un outil pour visualiser des coévolutions à l'échelle des transcrits, des gènes et des espèces.

Structure de la thèse

La thèse est organisée comme suit :

— L'introduction de la thèse présente le contexte dans lequel sont réalisés les travaux, puis les objectifs, la méthodologie utilisée, les contributions et la structure des chapitres suivants.

— Le chapitre 1 présente les notions de base en biologie et en évolution nécessaires pour mieux comprendre la suite du document.

— Le chapitre 2 effectue un état de l'art des méthodes de reconstruction et de correction des arbres phylogénétiques. Il met l'accent sur les arbres de gènes et les arbres de transcrits.
— Le chapitre 3 aborde la problématique de reconstruction d'arbres de gènes, d'arbres de transcrits et la segmentation d'ensembles de transcrits.
— Le chapitre 4 porte sur la simulation de l'évolution des séquences biologiques en tenant compte de l'épissage alternatif.
— Le chapitre 5 traite de la comparaison, de la segmentation non supervisée des séquences et de la reconstruction des arbres de gènes.
— Le chapitre 6 présente un outil interactif de visualisation de l'évolution conjointe des transcrits, des gènes et des espèces.
— La conclusion récapitule les contributions abordées dans cette thèse et les différentes perspectives d'application.

Chapitre 1

Notions de base en biologie et évolution

Le but de ce chapitre est de présenter les concepts de base en biologie et en évolution nécessaires à la compréhension de cette thèse. La première section définit par ordre d'inclusion les concepts de cellule, de chromosome, d'acide désoxyribonucléique (ADN), de gène, d'acide ribonucléique (ARN), de protéine, d'épissage alternatif, et de famille de gènes. La seconde section décrit les modèles d'évolution des espèces et des gènes. La dernière section présente quelques modèles et outils bio-informatiques de base pour l'alignement et la segmentation des séquences biologiques homologues.

1.1 Objets biologiques

1.1.1 Cellule

La cellule est la plus petite unité capable de remplir de façon autonome toutes les fonctions de la vie. Tout organisme vivant est constitué d'une ou plusieurs cellules. On distingue deux grands groupes d'organismes, à savoir : les Procaryotes et les Eucaryotes. Les Procaryotes sont des êtres unicellulaires, dépourvus de noyau et bordés d'une membrane. Les cellules des Eucaryotes sont généralement de plus grande taille, avec un noyau bordé d'une membrane. Indépendamment du type d'organisme, toutes

les cellules disposent d'une membrane plasmique, qui contient du cytoplasme et un matériel génétique sous forme d'acide désoxyribonucléique. La figure 1.1 illustre la structure d'une cellule des eucaryotes. Elle présente ses principaux composants dont le noyau qui contient le chromosome.

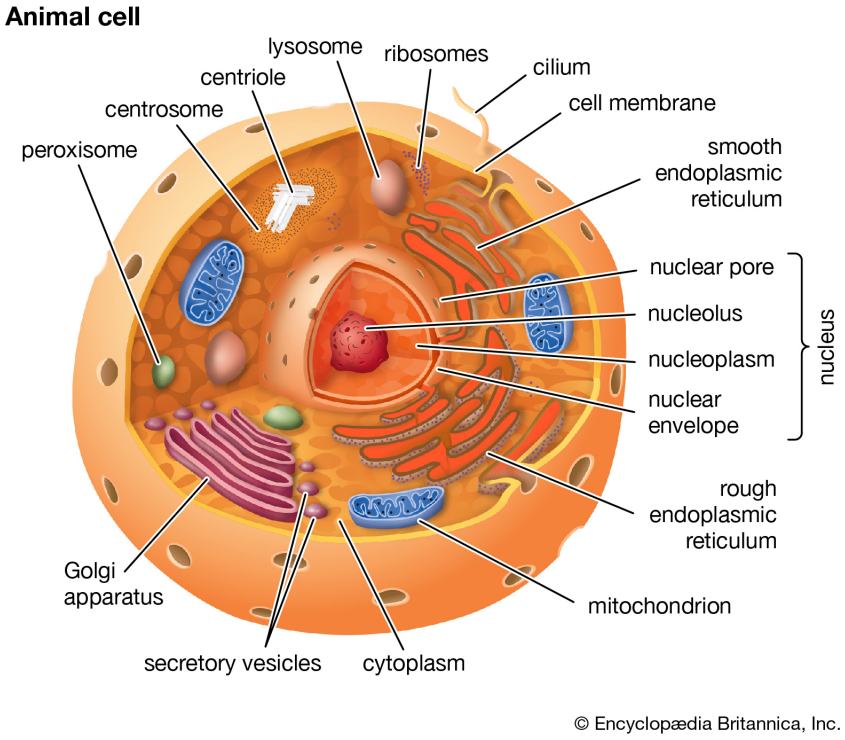


Figure 1.1 – Structure de la cellule des eucaryotes. Source : Encyclopædia Britannica, Inc.

1.1.2 Chromosome

Le chromosome, qui se trouve dans le noyau chez les cellules eucaryotes, est une structure cellulaire microscopique composée du matériel génétique. Tout être vivant dispose d'un nombre de chromosomes. Par exemple, la souris, le gorille et l'homme disposent respectivement de 20, 24 et 23 paires de chromosomes. La figure 1.2 illustre un chromosome extrait du noyau d'une cellule eucaryote.

1.1.3 ADN

L'ADN est une macromolécule qui une fois assemblée en une séquence nommée génome, porte toute l'information génétique d'un organisme, on parle alors de génome. Cette molécule encode des composants responsables de la production des ARN, des protéines, du métabolisme et de la reproduction des cellules. Somme toute, l'ADN permet la reproduction, le développement et le fonctionnement des êtres vivants. Concernant sa structure, l'ADN se présente comme un ensemble de deux brins enroulés l'un autour de l'autre pour former une double hélice comme présentée sur la figure 1.2 ci-contre. Chaque brin est composé d'une succession de quatre acides nucléiques. Ces nucléotides sont l'adénine (A), la cytosine (C), la guanine (G) ou la thymine (T). De façon plus formelle, l'ADN peut être défini comme un mot utilisant un alphabet composé de A, C, G et T. Un exemple de mot ADN serait la séquence : ATGGTGACAGAGTGGCAGAGTGCGACTTCTCCGAGTTGCTCG.

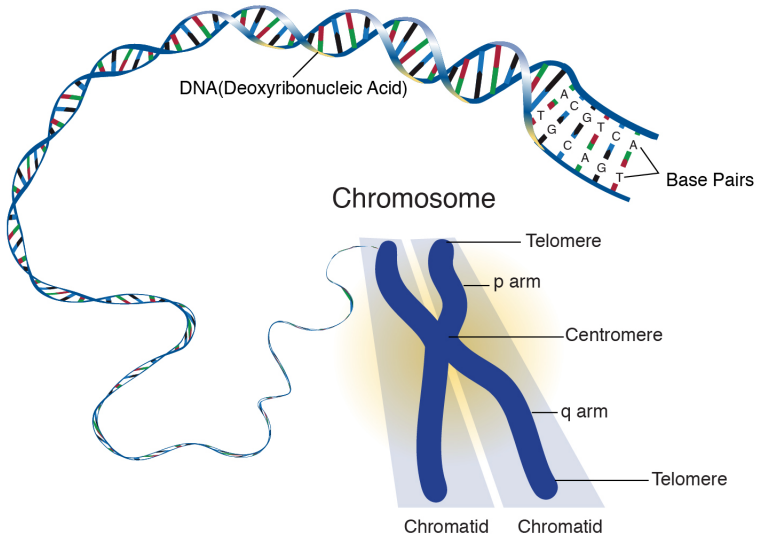


Figure 1.2 – Illustration d'une molécule d'ADN provenant du chromosome d'une cellule d'eucaryote. Source : National Human Genome Research Institute

1.1.4 Gène

Le gène est une portion de l'ADN qui encode des produits fonctionnels. C'est une composante de l'ADN qui contrôle partiellement ou totalement l'expression des caractéristiques chez tout être vivant. Le gène est aussi le support de l'évolution, car il est transmis aux générations suivantes avec des possibles mutations.

1.1.5 ARN

L'ARN est une séquence de nucléotides composée de l'adénine (A), de la guanine (G), de la cytosine (C) et de l'uracile (U). L'ARN est une copie d'un segment de brin d'ADN qui diffère de la séquence d'ADN par le fait que la thymine (T) est remplacée par l'uracile (U). L'ARN peut être lu par des ribosomes pour permettre la synthèse de protéines ; dans ce cas il est dit ARN codant. Il peut être directement impliqué dans des fonctions au sein de la cellule, dans cet autre cas il est dit ARN non codant. L'ARN provient de la transcription du gène. La figure 1.3 illustre la transcription d'un gène pour former un ARN. Un exemple d'ADN transcrit en ARN s'illustre comme suit :
- ADN
ATGGTGACAGAGTGGCAGAGTGCGACTTCTCCGAGTTGCTCG
ARN
AUGGUGACAGAGUGGCAGAGUGCGACUUCUCCGAGUUGCUCG

1.1.6 Épissage alternatif

L'épissage alternatif est le mécanisme utilisé par les cellules pour diversifier la production des ARN matures à partir de la séquence d'un même gène. Le mécanisme d'épissage permet à l'ARN de subir des étapes de coupures et ligation qui mènent à l'élimination de certains segments dont le résultat final est l'ARN mature. Les segments conservés sont appelés exons tandis que les segments éliminés par l'épissage sont appelés introns. L'épissage alternatif permet d'avoir différentes combinaisons d'exons que l'on nomme isoformes à partir d'un même gène. La figure 1.3 présente un ARN qui va subir l'épissage alternatif pour produire trois ARN matures. On distingue

cinq types d'épissage alternatif.

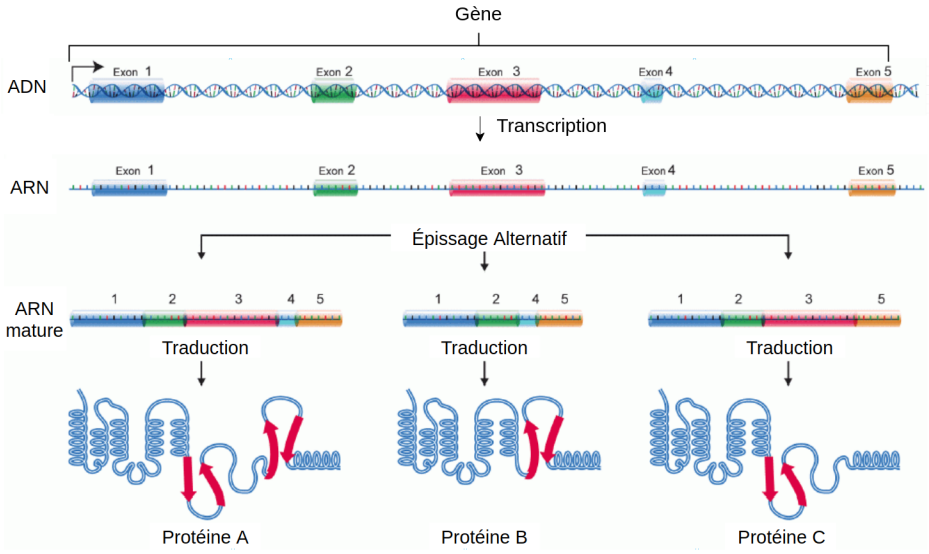


Figure 1.3 – Épissage alternatif permettant la production de trois protéines à partir d'un ARN. Source : *en.wikipedia.org/wiki/Alternative_splicing*

La figure 1.1.6 illustre les cinq mécanismes de l'épissage alternatif.

1. Extrémité 5 prime alternative : il existe plusieurs positions possibles de début du site d'épissage dans l'intron ;
2. Extrémité 3 prime alternative : il existe plusieurs positions possibles de fin du site d'épissage dans l'intron ;
3. Exon cassette : un exon est inclus dans un ARN mature et pas dans un autre ;
4. Exons mutuellement exclusifs : deux exons qui ne peuvent être simultanément inclus dans un ARN mature ;
5. Rétention d'introns : un intron est retenu pour faire partie d'un ARN mature.

1.1.7 Protéine

Les protéines sont l'une des molécules les plus abondantes dans les organismes vivants. Elles sont responsables d'un ensemble de fonctions parmi lesquelles la catalysation des réactions métaboliques, la réplication de l'ADN, le transport des molécules d'un endroit à un autre, etc.

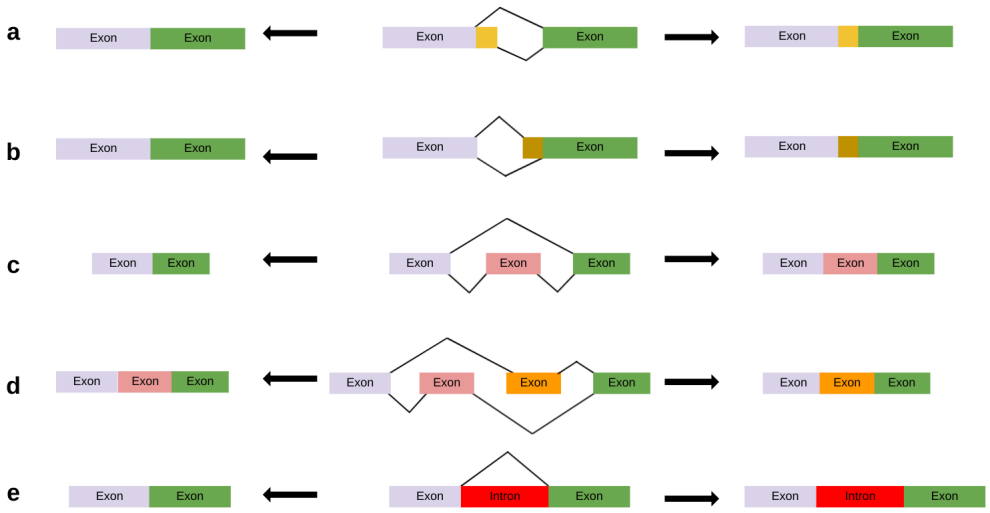


Figure 1.4 – I
llustration des cinq mécanismes de l'épissage alternatif. a) extrémité 5 prime alternatives, b) extrémité 3 prime alternatives, c) cassette exon, d) exons mutuellement exclusifs et e) rétention d'introns.

La protéine est à l'image de l'ADN, c'est un mot formé avec un alphabet de vingt acides aminés. Elle provient de la traduction de l'ARN codant par un processus biologique nommé traduction qui convertit chaque triplet d'ARN, appelé codon, en un acide aminé. L'ensemble des acides aminés et l'ensemble des règles de traduction sont donnés par le tableau 1.5. L'exemple suivant illustre une séquence d'ARN traduite en protéine :

>ARN mature

AUG GUG ACA GAG UGG CAG AGU GCG ACU UCU CCG AGU UGC UCG

>Protéine

M V T E W Q S A T S P S C S

1.1.8 Séquences codantes

Une séquence codante en anglais *CoDing Sequence (CDS)* est une région de l'ADN ou de l'ARN dont la séquence détermine la séquence d'acides aminés dans une pro-

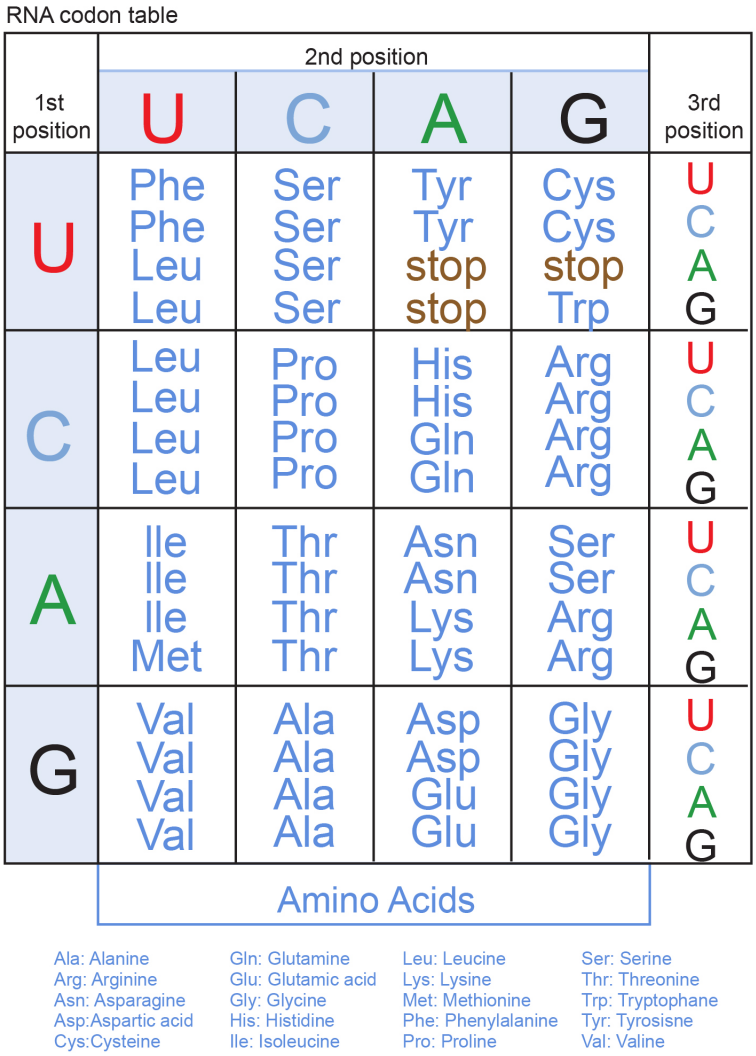


Figure 1.5 – Le code génétique standard. Source : National Human Genome Research Institute

téine.

1.2 Modèles d'évolution des espèces et des gènes

Les arbres phylogénétiques sont généralement utilisés pour représenter l'évolution d'un ensemble d'espèces, gènes ou alors d'une d'une séquences biologique. Cette présente les principaux arbres phylogénétiques.

1.2.1 Évolution des espèces

Un arbre d'espèces S sur un ensemble d'espèces $\mathcal{S}$ est un arbre enraciné dans lequel, les feuilles représentent les espèces de $\mathcal{S}$. Chaque nœud interne représente un événement de spéciation, qui correspond au moment où une espèce ancestrale se scinde en deux espèces descendantes génétiquement différentes et qui ne peuvent plus se reproduire ensemble. Le nœud racine est l'ancêtre commun de toutes les espèces qui sont aux feuilles. La figure 1.6 illustre un exemple d'arbre d'espèces.

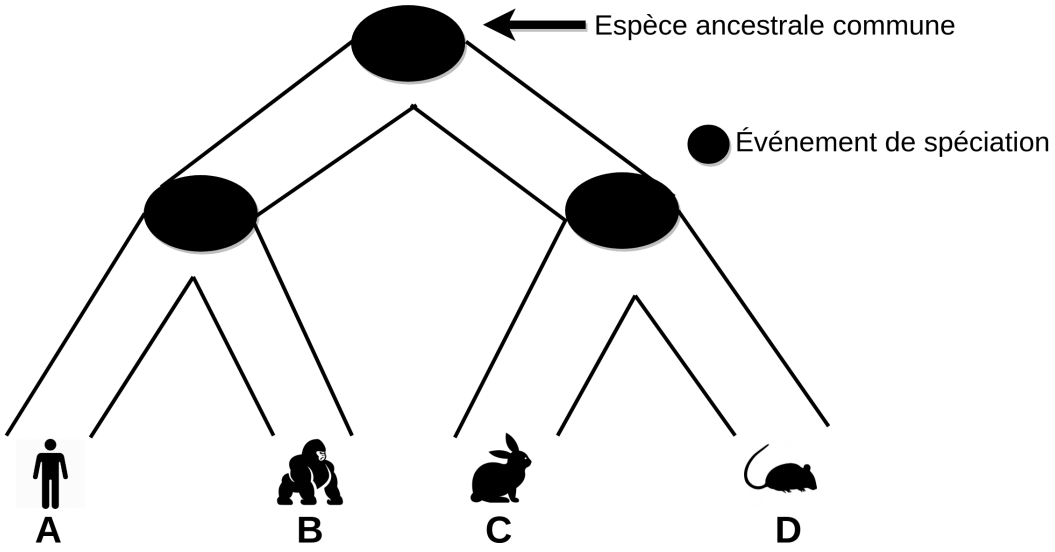


Figure 1.6 – Arbre phylogénétique de quatre espèces : aux feuilles on observe comme espèces l'humain, le chimpanzé, le lapin et la souris

1.2.2 Évolution des gènes

Un arbre de gènes G sur un ensemble de gènes $\mathcal{G}$ est un arbre enraciné dans lequel, les feuilles représentent les gènes de $\mathcal{G}$. Chaque nœud interne représente soit un événement de spéciation soit de duplication. Une spéciation de gène est induite par la spéciation d'espèce alors qu'une duplication de gène signifie que la copie d'un gène apparaît dans une même espèce. Le nœud racine est le gène ancestral commun de tous les gènes situés aux feuilles. La figure 1.7 illustre un arbre de gènes.

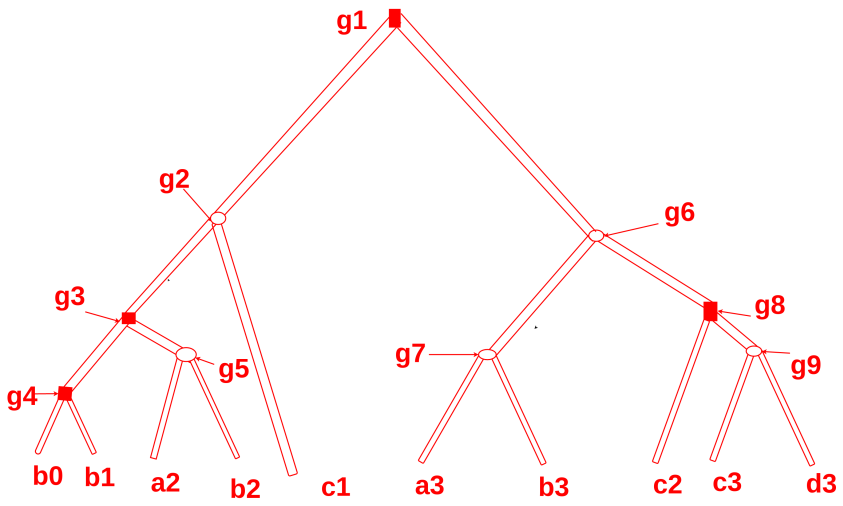


Figure 1.7 – Arbre de gènes sur un ensemble de dix gènes. Les carrés rouges représentent les nœuds de duplication, et les cercles blancs représentent les nœuds de spéciation. Les étiquettes des nœuds internes représentent les gènes ancestraux.

La notion d'arbre de gènes, est liée à celle de famille de gènes. Une famille de gènes est un ensemble de gènes qui descendent d'un ancêtre commun au travers d'événements de spéciations et duplications. Dans l'exemple de la figure 1.7, l'ensemble de gène $\{b0, b1, a2, b2, c1, a3, b3, c2, c3, d3\}$ constitue une famille de gène. Les gènes d'une même famille de gène sont dits homologues. De façon plus générale, deux séquences sont dites homologues si elles descendent d'une même séquence ancestrale. Deux gènes sont paralogues (respectivement. orthologues) lorsque leur plus proche ancêtre commun dans l'arbre des gènes a subi une duplication (respectivement spéciation). Par exemple $(b0, b1)$, $(b0, a2)$, $(b0, b2)$ sont trois exemples de gènes paralogues. $(b0, c1)$, $(b1, c1)$, $(a2, b2)$ sont trois exemples de gènes orthologues.

1.2.3 Comparaison d'arbres

Le AU test (*approximately unbiased test*) [59] est une méthode qui permet de comparer un ensemble d'arbres. Elle consiste à comparer la vraisemblance de chaque arbre qui est calculée à partir du bootstrap à plusieurs échelles. Par la suite, en se basant sur la comparaison des ces valeurs de vraisemblance obtenues, les arbres ayant un faible support statistique sont rejetés.

1.3 Identification et représentation des séquences homologues

1.3.1 Alignement de séquences biologiques

Étant donné un ensemble de séquences d'ADN, de séquences d'ARN ou de séquences de protéines, un alignement de ces séquences permet de les superposer dans le but d'identifier les sous-chaînes conservées et de maximiser le score de similarité entre les séquences. L'alignement des séquences est par la suite utilisé d'une part pour identifier les régions conservées au cours de l'évolution depuis une séquence ancestrale commune, et d'autre part pour la reconstruction des arbres phylogénétiques. Dans un alignement de séquences, les résidus d'une colonne peuvent être identiques. On parlera alors de résidus conservés *(match)*. Si les résidus sont différents, on parlera de mutation *(mismatch)*. On peut finalement observer parmi les résidus le caractère *"-"* qui symbolise l'insertion ou la délétion d'un résidu. Lorsque le nombre de séquences à aligner est égal à deux, on parle d'alignement par paire. Si ce nombre est strictement supérieur à deux, on parle d'alignement multiple. La figure 1.8 illustre un exemple d'alignement multiple de séquences. Dans la littérature, il existe trois grands types d'alignement de séquences. Chacun de ces types a des spécificités qui sont en compétition les unes avec les autres dans des contextes spécifiques.

Le premier type d'alignement est l'alignement global résolu par l'algorithme de Needleman et Wunsch[46]. Il est utilisé pour aligner les séquences sur toutes leurs longueurs. Il est généralement utilisé quand les séquences alignées sont de longueur similaire. Cet algorithme peut être utilisé pour aligner les séquences illustrées sur la

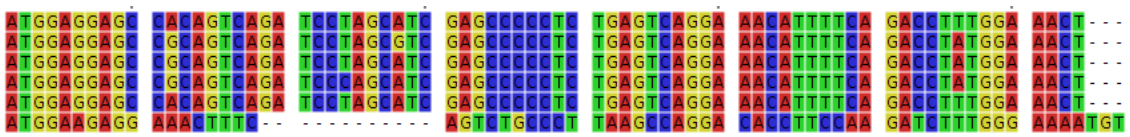


Figure 1.8 – Illustration de l'alignement multiple. On observe ici des colonnes dans lesquelles les résidus sont tous conservés et qui ont la même couleur. Les colonnes aux couleurs différentes indiquent la présence des mismatch.

figure 1.8 car ces séquences ont des longueurs similaires.
Le second type d'alignement est l'alignement local, celui de Smith et Waterman[61], qui est utilisé pour rechercher et aligner uniquement les motifs les plus conservés entre les deux séquences.
Le troisième et dernier type est l'alignement semi-global, utilisé quand une des séquences est incluse dans l'autre ou quand on s'autorise des insertions et délétions aux extrémités qui ne seront pas prises en compte dans le score de l'alignement.

1.3.2 Regroupement des séquences biologiques

Étant donné un ensemble de séquences (ADN, ARN ou protéines), le regroupement est une opération visant à répartir les séquences dans des groupes dont chacun comporte des séquences similaires et différentes des autres. En d'autres termes, on cherche à cerner un ensemble de groupes ayant un maximum de similarité entre eux tout en minimisant la similarité entre les séquences de deux groupes. Ces groupes constituent des ensembles de séquences homologues qui sont utilisés par la suite pour prédire de nouvelles fonctions biologiques ou de nouveaux gènes, ou pour la reconstruction d'arbres phylogénétiques. On distingue deux types de méthodes de regroupement de séquences : le regroupement hiérarchique et le regroupement par partitionnement.

Les méthodes par regroupement hiérarchique sont les plus utilisées dans le contexte des séquences biologiques. Ceci est dû au fait que cette approche explique naturellement les relations évolutives entre les données. Le regroupement hiérarchique peut être soit ascendant soit descendant. Dans le cas ascendant, on suppose au départ que chaque séquence forme un groupe, puis de manière itérative, on fusionne les deux groupes les plus proches jusqu'à ce qu'il ne reste plus qu'un seul groupe. Le regrou-

pement descendant débute avec tous les objets d'un seul groupe. Puis, toujours de manière itérative, choisit un groupe et le divise en deux sous groupes tant que cela améliore le regroupement. En guise d'exemple d'algorithmes de regroupement hiérarchique, on peut citer ProClust[52], CLUSS[34], BlastClust[16], TransClust[68, 69].

Le regroupement par partitionnement est de plus en plus utilisé : il se popularise davantage avec l'arrivée des méthodes telle que *K-means* dans les années 1967. Il permet de grouper les séquences en un ensemble de partitions. Ici, le nombre de groupe est connu, il est égal à K. Dans la majorité des cas, cette approche cherche à minimiser la somme des distances moyenne des séquences de chaque groupe au centre du groupe. Les méthodes les plus utilisées en phylogénétique sont OrthoMCL[42], Orthofinder[19] et ProteinOrtho[37].

1.4 Conclusion

Dans ce chapitre il était question d'introduire des notions de base en biologie et en évolution nécessaires pour mieux comprendre cette thèse. Dans la première partie, nous avons présenté les principaux objets biologiques tels que la cellule, le chromosome, l'ADN, le gène, l'ARN, l'épissage alternatif, les protéines. Par la suite, nous avons présenté des modèles d'évolution d'espèces et de gènes. Enfin nous avons présenté deux types d'outils bio-informatique très utilisés pour l'analyse de séquences biologiques homologues à savoir l'alignement et le regroupement des séquences. Dans le chapitre suivant, nous présenterons l'état de l'art des méthodes de reconstruction et de correction d'arbres phylogénétiques.

Chapitre 2

Construction et correction d'arbres de gènes

Dans ce chapitre nous traitons de la problématique de reconstruction d'arbres phylogénétiques. La première section présente l'utilité des arbres de gènes et des arbres de transcrits. Ensuite, l'état de l'art des méthodes de reconstruction des arbres phylogénétiques est présenté. Par la suite, nous présentons les méthodes de construction d'arbres de gènes, enfin, nous présentons les méthodes de correction d'arbres de gènes.

2.1 Utilité des arbres de gènes et de transcrits

De nos jours, les méthodes de reconstruction de l'histoire évolutive des organismes vivants sont de plus en plus basées sur des données moléculaires. Ceci nous permet d'aller au-delà des caractères morphologiques pour tirer profit de nos connaissances sur les séquences biologiques. Les études phylogénétiques ont rendu possibles plusieurs découvertes en biologie. La phylogénie est utilisée pour l'annotation fonctionnelle des gènes, la prédiction et la classification des gènes de façon fiable. Elle permet par exemple d'identifier l'origine d'un pathogène. En bio-informatique, la phylogénie est utilisée comme étape de base dans plusieurs types d'analyse. Ainsi donc, la précision de ces analyses dépend de celle de la phylogénie utilisée.

La figure 2.1 illustre à grande échelle trois étapes nécessaires à la reconstruction

des arbres phylogénétiques.

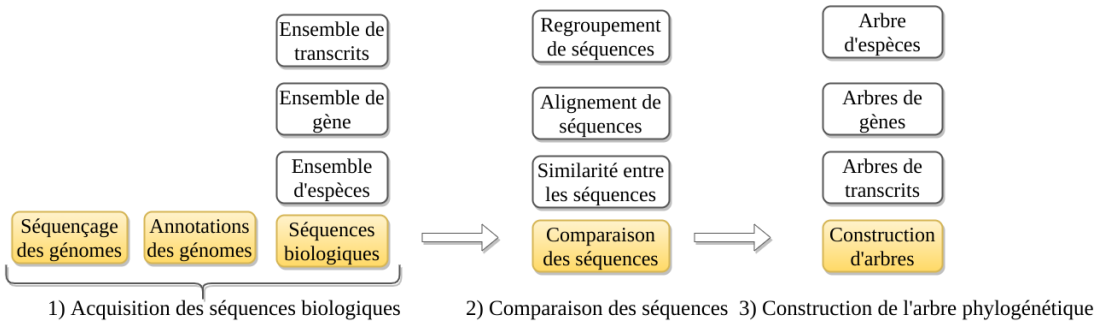


Figure 2.1 – Principales étapes pour la reconstruction des arbres phylogénétiques

La première étape concerne l'acquisition des séquences biologiques dont on souhaite reconstruire l'évolution. Ces données biologiques qui proviennent des *reads* sont séquencées et regroupées en famille. Une famille de gènes est un ensemble de gènes qui descendent d'une même séquence ancestrale. Les arbres phylogénétiques sont estimés à différents niveaux d'évolution donc les principaux incluent : les espèces[12, 70, 2], les gènes [45, 35, 66], les transcrits[35, 10, 11], les domaines[45, 41, 40] et les syntenies[54, 17]. Dans la suite de cette thèse, l'étape d'acquisition des données correspondra toujours à l'acquisition des séquences d'une famille de gènes et son ensemble de transcrit. Sauf mention particulière, nous supposons que nous disposons toujours de ces séquences.

La seconde étape implique une représentation préliminaire permettant d'analyser les séquences biologiques sous forme de données catégorielles. La représentation la plus courante est l'alignement des séquences. Elle permet par la suite de définir des mesures de similarité entre paires de séquences pour les regrouper selon les scores de similarité obtenus.

Enfin la troisième étape qui est celle de la construction des arbres peut être réalisée de diverses manières. Dans le cas de la reconstruction d'arbre de gènes et d'espèces, la figure 2.2 extraite de [65] présente les différents scénarios de reconstruction des arbres de gènes et d'espèces.

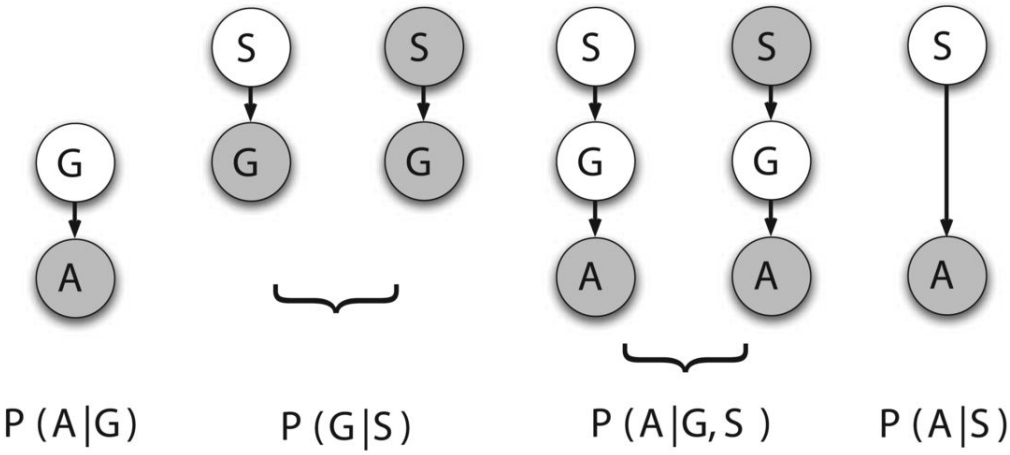


Figure 2.2 – Différents scénarios de reconstruction des arbres de gènes et d'espèces. Les nœuds gris sont considérés comme connus et les nœuds blancs sont déduits. Cette figure montre différents problèmes d'inférence d'arbres phylogénétiques en considérant les alignements de gènes, les arbres des gènes, les arbres d'espèces ou plusieurs d'entre eux comme des données. A représente l'alignement des gènes, G représente l'arbre de gènes et S représente l'arbre d'espèces. $P(X|Y)$ désigne le fait de chercher Y sachant X

La section suivante présente les principales méthodes utilisées pour la reconstruction des arbres de gènes.

2.2 Approches de construction d'arbres phylogénétiques

Il existe plusieurs méthodes pour la reconstruction des arbres phylogénétiques. Chacune d'elles ayant sa spécificité. On distingue les méthodes basées sur distances entre les séquences aux feuilles, les méthodes de parcimonie les méthodes probabilistes.

2.2.1 Méthodes basées sur les distances entre les gènes

Ces méthodes utilisent comme unique entrée une matrice de similarité entre les séquences.

Unweighted Pair Group Method with Arithmetic Mean (UPGMA)

UPGMA[62] est un algorithme permettant, à partir d'une matrice de distance de construire un arbre enraciné. Cette méthode est simple et intuitive. Les principales étapes de UPGMA sont :

1. L'étape initiale consiste à définir chaque séquence comme étant une feuille de l'arbre finale, telle que chaque séquence représente un groupe.
2. De façon itérative, jusqu'à ce qu'il ne reste plus qu'un groupe :
   — Ajouter un nouveau groupe C_k regroupant les deux groupes C_i et C_j les plus proches et retirer ces deux groupes de la liste des groupes.
   — Ajouter dans l'arbre un nouveau nœud correspondant au nouveau groupe, comme parent des deux noeuds regroupés.
   — Pour chaque groupe existant, la distance entre deux groupes C_i et C_j est définie comme suit : $d_{ij} = \frac{1}{|C_i||C_j|} \sum_{p \in C_i, q \in C_j} d_{pq}$. Si $C_k = C_i \cup C_j$ la distance entre C_k et un autre groupe C_l est donnée par la formule récursive suivante : $d_{kl} = \frac{d_{il}|C_i| + d_{jl}|C_j|}{|C_i| + |C_j|}$.

UPGMA construit un arbre enraciné des feuilles vers la racine, en ajoutant un nœud à chaque itération. Les distances de toutes les feuilles à la racine de l'arbre binaire obtenu par UPGMA sont identiques, ce qui signifie que les vitesses d'évolutions sont les mêmes dans toutes les lignées. Pourtant, les séquences qui sont aux feuilles peuvent provenir d'espèces différentes avec des vitesses d'évolution différentes sur les branches. Ceci constitue donc une limite importante à l'utilisation de UPGMA dans le cadre de la reconstruction des phylogénies.

Neighbour Joining (NJ)

NJ[56] est un algorithme dédié à la construction d'arbres phylogénétiques qui tiennent compte des différentes vitesses d'évolution sur les branches de l'arbre. La prise en compte des vitesses d'évolution implique la notion d'arbre additif. Étant donnée une matrice de distance symétrique D entre les n feuilles d'un arbre, cet arbre est dit additif si ses arcs sont étiquetés avec des distances de sorte que pour chaque paire de feuilles (i, j) dans l'arbre, la somme des distances des arêtes du chemin de i à j est égale à $D(i, j)$.

La méthode NJ se base sur l'hypothèse qu'il existe un arbre additif pour la matrice de distance donnée comme entrée, et produit un tel arbre non enraciné. Les principales étapes de NJ sont :

1. L'étape initiale consiste à définir chaque séquence comme étant une feuille, telle que chaque séquence représente un groupe.
2. De façon itérative, jusqu'à ce qu'il ne reste plus qu'un groupe :
   — Ajouter un nouveau groupe C_k regroupant les deux groupes C_i et C_j minimisant la formule $d_{ij} - (r_i + r_j)$ avec $r_i = \frac{1}{|L| - 2} \sum_{k \in L} d_{ik}$, et retirer ces deux groupes de la liste des groupes.
   — Ajouter dans l'arbre un nouveau nœud correspondant au nouveau groupe C_k, comme parent des deux noeuds correspondant aux groupes retirés, de sorte que les nouvelles arêtes de l'arbre sont étiquetées $d_{ik} = \frac{1}{2}(d_{ij} + r_i - r_j)$ et $d_{jk} = d_{ij} - d_{ik}$.
   — Pour chaque autre groupe C_m, recalculer la distance entre C_m et C_k suivant la formule : $d_{km} = \frac{1}{2}(d_{im} + d_{jm} - d_{ij})$

La méthode NJ construit un arbre non enraciné. Pour enraciner cet arbre, il suffit d'ajouter une séquence extérieure au groupe des autres séquences considérées (*out-group*). La position du branchement de cette séquence sur l'arbre indique la position de la racine de l'arbre. Une autre stratégie d'enracinement d'arbre est de considérer comme racine le milieu d'un plus long chemin dans l'arbre entre les feuilles (hypothèse de l'horloge moléculaire qui stipule que les mutations génétiques s'accumulent dans un génome à une vitesse constante).

2.2.2 Méthodes de parcimonie

Étant donné un alignement de séquences, les méthodes par parcimonie ont pour but de trouver un arbre qui minimise le nombre total de modifications évolutives (substitutions, délétions, ou insertions de résidus) pour passer d'une séquence à l'autre sur les branches de l'arbre.

Les principales étapes des méthodes de parcimonie sont :

1. Calculer un alignement multiple des séquences (gènes)
2. Pour chaque colonne de l'alignement, trouver un arbre minimisant le nombre de modifications évolutives sur les branches de l'arbre. La recherche de l'arbre optimal se fait par énumération des arbres ou par l'utilisation des heuristiques d'exploration de l'espace de recherche qui évite d'énumérer tous les arbres.
3. À partir de l'ensemble des arbres obtenus, trouver un super-arbre (arbre obtenu en combinant plusieurs sous arbres.) qui minimise la somme totale des nombres de modifications évolutives pour toutes les colonnes de l'alignement.

Les deux principaux algorithmes permettant de calculer le nombre de modifications évolutives induites par un arbre sont l'algorithme de Fitch[20] et celui de Sankoff[15].

Algorithme classique de Fitch

Étant donné un arbre phylogénique T dont les feuilles sont des résidus, le principe de l'algorithme de Fitch est d'associer à chaque nœud interne un ensemble d'états menant à une complétion optimale tout en évaluant le nombre minimum de mutations nécessaires dans une configuration. Par la suite, l'arbre choisi est celui qui minimise le nombre de mutations.

Le principe de l'algorithme classique de Fitch est le suivant :

1. Initialiser le nombre de modifications C à 0.
2. Pour chaque nœud k de l'arbre, en allant des feuilles vers la racine (parcours postfixe des nœuds) :
   - Si k est une feuille, poser $R_k = \{\text{étiquette de } k\}$
   - Si k n'est pas une feuille,
     - Calculer $Inters = R_i \cap R_j$, où i, j sont les enfants de k ;
     - Si $Inters == \emptyset$, poser $R_k = R_i \bigcup R_j$ et incrémenter C de 1
     - Sinon, poser $R_k = Inters$;
   - Fin : le poids minimal de l'arbre est C.

Parcimonie pondérée de Sankoff

L'algorithme de parcimonie pondérée de Sankoff est plus général que celui de Fitch. Il ne calcule pas juste le nombre de mutations, mais considère également un poids $S(a;b)$ pour la substitution d'une lettre a par b. Ce poids peut être utilisé pour donner plus de poids aux transitions entre les nucléotides, et moins de poids aux transversions entre nucléotides. Il étiquette les nœuds internes de l'arbre de sorte à minimiser le poids total de l'arbre. L'étiquetage des nœuds est réalisé par récurrence des feuilles à la racine, en calculant l'étiquette d'un nœud à partir des étiquettes de ses nœuds enfants. $S_k(a)$ désigne le poids du sous-arbre de racine k, sous la condition que k est étiqueté par a.

Les principes de l'algorithme de parcimonie pondérée de Sankoff sont :

1. Initialiser k le numéro de la racine à $2n-1$, n étant le nombre de feuilles.
2. Par récurrence, calculer $S_k(a)$ pour tous les a :
   - Si k est une feuille,
     - poser $S_k(a) = 0$ pour a étiquette de k, $S_k(a) = \inf$ sinon ;
   - Si k n'est pas une feuille,
     - Calculer $S_i(a), S_j(a)$ pour tous les a, où i, j sont des enfants de k ;
     - Poser $S_k(a) = min_b(S_i(b) + S(a,b)) + min_b(S_i(b) + S(a,b))$
   - Fin : le poids minimal de l'arbre est $min_a S_{2n-1}(a)$.

2.2.3 Méthodes de maximum de vraisemblance

L'approche par le maximum de vraisemblance[48] est une approche probabiliste permettant de trouver l'arbre le plus probable d'avoir généré les séquences à ses feuilles. Les hypothèses de cette méthode sont : le processus de substitution d'une séquence en une autre suit un modèle probabiliste dont on connaît l'expression mathématique. Les sites (nucléotides) évoluent indépendamment les uns des autres ; ils évoluent selon la même loi. Les taux de substitution ne changent pas au cours du temps le long d'une branche. Ils peuvent cependant varier entre branches. La proba-

bilité d'un arbre est donnée par la formule suivante

$$L = P(D|T) = \prod_{i=1}^{m} P(D^i|T). \quad (1)$$

D représente les données, T l'arbre pour lequel nous calculons la probabilité et m le nombre de sites (nucléotides) de notre séquence d'ADN. Posons $L_i = P(D^i|T)$, l'équation 1 peut donc s'écrire sous la forme :

$$L = L_1 \times L_2 \times L_3 \times ... \times L_m = \prod_{i=1}^{m} (L_i).$$

où L_i est la vraisemblance pour le nucléotide à la position i. Ce produit devient une somme lorsque nous prenons le logarithme naturel de L :

$$\log L = \log L_1 + \log L_2 + \log L_3 + ... + \log L_m = \sum_{i=1}^{m} \log L_i.$$

De cette manière, il devient possible d'attribuer une probabilité à tous les arbres. L'arbre ayant la plus grande probabilité manifeste le maximum de vraisemblance, c'est donc le plus probable.

2.2.4 Méthode d'inférence bayésienne

La méthode d'inférence bayésienne se base sur le théorème de Bayes pour calculer la probabilité d'un arbre a *posteriori*. Cette méthode permet de choisir parmi un ensemble d'arbres, celui ayant la plus grande probabilité. De façon formelle, $P(T|S) = \frac{P(A|T)P(T)}{P(S)}$. $P(T|S)$ représente la probabilité à *posteriori* de l'arbre T sachant les séquences S. $P(T)$ est la probabilité d'obtenir l'arbre T qui vaut 1 divisé par le nombre d'arbres. $P(S)$ est la probabilité d'obtenir l'ensemble des séquences S. $P(S|T)$ est la probabilité d'obtenir l'ensemble des séquences S étant donné T. Les méthodes d'inférence bayésienne se sont popularisées avec l'arrivée des grandes ressources de calcul et l'intégration des méthodes de Monte-Carlo par chaînes de Markov (MCMC). L'algorithme Metropolis-Hastings est l'une des méthodes de MCMC les plus couramment utilisées. Elle l'est largement pour échantillonner de manière

aléatoire des probabilités de distribution complexes et multidimensionnelles. L'idée générale de cet algorithme est d'explorer l'espace des arbres de manière à obtenir une distribution stationnaire des arbres qui représente la distribution a *posteriori*. L'espace des arbres peut être imagé sous la forme d'un grand graphe regroupant tous les arbres connectés les uns aux autres. En supposant que nous connaissons la distribution des arbres $f(t)$, les principales étapes de l'algorithme de Metropolis-Hastings sont :

1. Choisir aléatoirement un arbre initial t_i.
2. Choisir un second arbre t_j proche de t_i.
3. Calculer le ratio d'acceptation comme suit : $R = f(t_j)/f(t_i)$
4. À partir de la distribution uniforme (0,1), générer un nombre aléatoire u.
   — si $u \leq R$, $t_{i+1} = t_j$
   — sinon $t_{i+1} = t_i$
5. À ce stade, le processus est répété de l'étape 2 à 4 N fois. Le nombre de fois qu'un arbre T est visité par la chaîne MCMC sur le nombre total d'itérations représente une approximation non biaisée de $P(T_i|D)$. N est fixé car il est fort possible que l'algorithme n'atteigne pas une distribution d'équilibre.

Cet algorithme nécessite qu'il soit possible et facile d'échantillonner des arbres.

2.3 Méthodes courantes de reconstruction d'arbres basées sur les super-arbres

Dans le souci d'estimer des arbres de gènes de façon fiable, de nombreuses méthodes ont été développées. Dans la suite de cette section, on présente sommairement les méthodes de super-arbres, les méthodes de réconciliation, les méthodes qui combinent la réconciliation et les super-arbres et enfin, les méthodes qui combinent la réconciliation et la vraisemblance.

2.3.1 Approches par super-arbres

Les méthodes de super-arbres ont été largement utilisées pour reconstruire les arbres d'espèces. Ici, nous nous intéressons à leur usage pour la reconstruction des arbres de gènes. Dans ces approches, on prend en entrée une forêt d'arbres F qui peuvent avoir des feuilles en commun. Dans le cas des gènes, les arbres de F sont par exemple des arbres d'orthologues, c'est-à-dire des arbres n'ayant aucun noeud de duplication. Le problème de super-arbres consiste à combiner cette forêt d'arbres en un seul arbre phylogénétique contenant toutes les feuilles de tous les arbres de cette forêt.

Super-arbres par consensus

Les méthodes de super-arbres par consensus supposent que tous les arbres de la forêt d'arbres ont le même ensemble de feuilles et que cette forêt d'arbres peut être inconsistante. Cette méthode retourne un arbre représentatif de cette forêt d'arbre. Les méthodes de consensus les plus utilisées sont listées dans la suite.

Consensus strict[43] est la méthode de consensus la plus naturelle de toutes les méthodes de consensus. Étant donné F, l'arbre consensus strict contient uniquement des clades (groupe monophylétique) qui sont communs à tous les arbres présents dans F. Le point positif du consensus strict est qu'il préserve tous les clades qui sont présents dans les arbres à fusionner. La figure 2.3 illustre deux arbres t1 et t2 et le consensus strict de ces deux arbres. On note deux clades communs à ces deux arbres : (a, b, c, d) et (a, b, c).

Loi de la majorité (*majority rule*) est une méthode calculant l'arbre dont les clades sont présents dans plus de 50% des arbres de F. La figure 2.4 illustre la loi de la majorité sur trois arbres $t1$, $t2$ et $t3$. Les clades qui apparaissent dans plus de la moitié des trois arbres sont $\{a, b\}, \{a, b, c\}$ et $\{a, b, c, d\}$

Arbre de consensus glouton (*Greedy consensus tree*) est une extension de la loi de la majorité. Cette approche consiste à construire de manière itérative un ensemble de clades compatibles. L'ensemble des clades apparaissant dans les arbres de F est trié par ordre décroissant en fonction de leur fréquence. On construit par la suite un ensemble de clades compatibles S itérativement. Le premier clade ayant

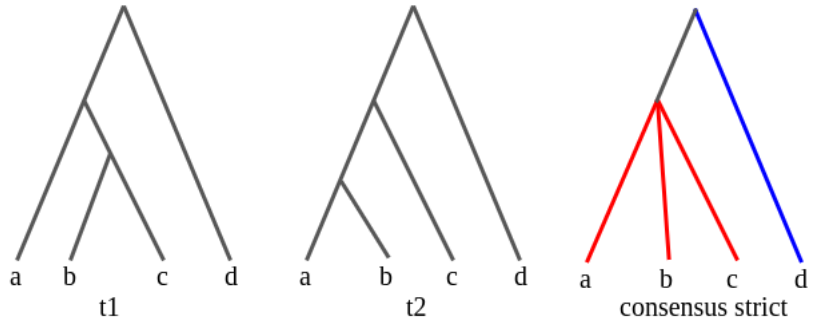


Figure 2.3 – Consensus strict de deux arbres. (a, b, c) et (a, b, c, d) représentent les deux clades commun à t1 et t2.

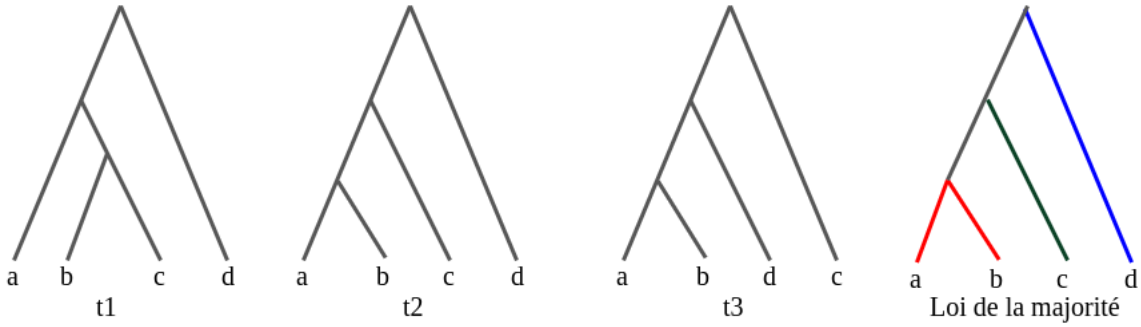


Figure 2.4 – La loi de la majorité de trois arbres t1, t2 et t3. Les clades majoritaires sont $\{a, b\}, \{a, b, c\}$ et $\{a, b, c, d\}$

la plus grande fréquence est inséré dans S. Les clades restants sont considérés dans l'ordre décroissant : si un clade est compatible avec tous les clades déjà dans S, alors il est inclus dans S. À la fin du processus, nous obtiendrons un ensemble de clades correspondant à un arbre phylogénétique. Cela donne l'arbre de consensus glouton pour F.

Arbre MAST (*Maximum Agreement SubTree*) consiste à trouver le plus grand sous-arbre en accord avec tous les arbres de F. De façon formelle, le MAST T est un arbre tel que : l'ensemble des feuilles de T est un sous-ensemble des feuilles des arbres appartenant à F, et, pour chaque arbre T_i de F, l'arbre obtenu en supprimant dans T_i tous les noeuds qui ne sont pas dans T est un arbre qui est identique à T. Une fois l'arbre MAST obtenu, il peut être complété avec les feuilles manquantes pour obtenir un arbre consensus.

Une des grandes critiques des méthodes de super-arbres pour la reconstruction des phylogénies des séquences provenant de différentes espèces, gènes, familles est qu'elle est guidée par la comparaison et la combinaison des topologies d'arbres plutôt que sur les critères d'inférence phylogénétique tel que la comparaison du nombre d'événements d'évolution induits par l'arbre reconstruit. Dans le chapitre suivant, nous introduirons un ensemble de méthodes qui exploitent d'une part l'information sur la proximité des espèces auxquelles appartiennent les gènes et d'autre part la vraisemblance des séquences aux feuilles pour reconstruire de meilleurs arbres de gènes.

2.4 Méthodes courantes de construction des arbres de gènes basées sur la réconciliation

2.4.1 Définitions de la réconciliation

La réconciliation : est une fonction s de l'ensemble des sommets d'un arbre de gènes G vers l'ensemble des sommets d'un arbre d'espèces S, tel que chaque sommet de G a une et une seule image dans S, et les relations d'ancestralité sont préservées. On en déduit un étiquetage des sommets de G comme suit : un sommet x est une duplication si x a la même image qu'un de ses enfants, sinon, c'est une spéciation. La figure 2.5 illustre une réconciliation entre un arbre de gènes et un arbre d'espèces.
Coût de réconciliation : Basé sur la réconciliation entre un arbre de gènes et un arbre d'espèces, le coût de réconciliation est le nombre d'opération de duplication et ou de perte de gènes qu'il faut effectuer pour réconcilier un arbre de gènes à un arbre d'espèces.

L'arbre de gènes G qui sera retourné est un arbre binaire enraciné dont les noeuds internes sont soit des noeuds de duplication soit des noeuds de spéciation. On observe également des événements de pertes de gènes. Selon le problème que l'on étudie, on peut choisir comme coût de réconciliation le nombre de duplications, le nombre de pertes ou la somme des deux. On peut également pondérer chacun des scores pour avoir un modèle de score plus flexible. La figure 2.5 illustre un exemple d'arbre de gène réconcilié dans un arbre d'espèces. Le coût de réconciliation est 2, soit 1 duplication

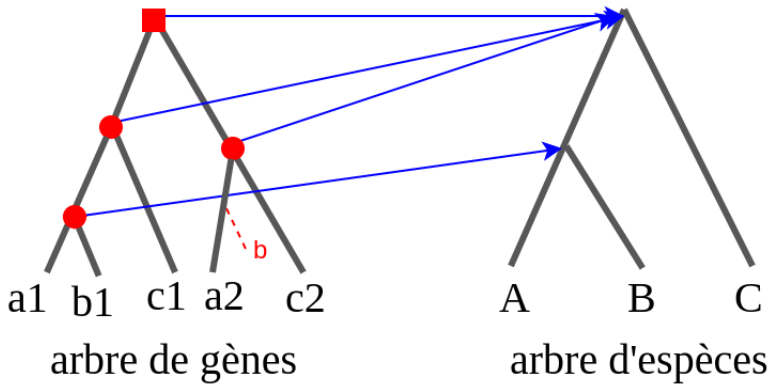


Figure 2.5 – Réconciliation entre un arbre de gènes et un arbre d'espèces. Le carré rouge représente un nœud de duplication d'un gène, tandis que les cercles rouges représentent des nœuds de spéciation de gènes et la ligne pointillée représente une perte de gène. A, B et C représentent trois espèces. L'espèce A possède deux gènes a1 et a2, l'espèce B possède un gène b1, l'espèce C possède deux gènes c1 et c2.

+ 1 perte si on suppose que le coût de réconciliation = nombre de duplications + le nombre de pertes.

2.4.2 Construction des arbres de gènes basée sur la réconciliation

Dans les méthodes de super-arbres, on suppose que l'on a une forêt d'arbres qui est fournie et que l'on souhaite fusionner. Dans le cas des approches basées sur la réconciliation[66, 23, 22, 65], on suppose que l'on a un ensemble de gènes et un arbre d'espèces. On cherche l'arbre de gènes dont le coût de réconciliation avec l'arbre d'espèces est minimal. Le problème de réconciliation peut être formulé ainsi :

Problème de réconciliation :
Entrée : Un arbre d'espèces S pour $\mathcal{S}$;
Une famille de gène $\mathcal{G}$ tel que $\{s(x) : x \in \mathcal{G}\} = \mathcal{S}$.
Sortie : Un arbre de gènes G pour $\mathcal{G}$ qui minimise le coût de réconciliation $X(G, S)$ entre G et S où $X(G, S) = D(Nombrededuplication)$, ou $P(Nombredeperte)$, ou $D + P$.

2.4.3 Construction des arbres de gènes basée sur la réconciliation et les super-arbres

Afin de tirer avantage de la possibilité d'obtenir un ensemble d'arbres sur un sous-ensemble de gènes orthologues, les méthodes qui sont basées sur les approches de réconciliation et super-arbres[36, 35] cherchent le super-arbre qui combine cet ensemble d'arbres tout en ayant un coût minimal de réconciliation. Le problème peut être formulé ainsi :

Extension du problème de super-arbre et de réconciliation :

Entrée : Un arbre d'espèce S pour $\mathcal{S}$;

Une famille de gène $\mathcal{G}$; telle que $\{s(x) : x \in \mathcal{G}\} = \mathcal{S}$

Une forêt de sous-arbres avec chevauchement F ;

Sortie : Un super-arbre de gène G qui combine tous les arbres de F et ayant le plus petit coût de réconciliation $X(G, S)$ entre G et S.

L'avantage de cette méthode est que le super-arbre final G conserve les clades les plus importants qui sont présents dans les arbres de F. Mais aussi, le nombre de duplications et de pertes est minimisé.

2.4.4 Construction des arbres de gènes basée sur la réconciliation et la vraisemblance

À l'instar des méthodes qui combinent l'approche de super-arbres et la réconciliation, les méthodes basées sur la réconciliation et la vraisemblance[47, 57] combinent l'approche par réconciliation et la vraisemblance des séquences aux feuilles dans le but de trouver un arbre de gènes qui minimise le coût de réconciliation et maximise la vraisemblance des séquences.

Construction des arbres de gènes basée sur les méthodes basées sur la réconciliation et la vraisemblance :

Entrée : Un alignement D des gènes d'une famille de gène ;

Un arbre d'espèce S pour $\mathcal{S}$; tel que $\{s(x) : x \in \mathcal{G}\} = \mathcal{S}$.

Sortie : Un arbre de gène G sur $\mathcal{G}$ qui maximise la vraisemblance des séquences aux feuilles et qui minimise le score de réconciliation de G avec S.

2.5 Corrections d'arbres de gènes

Il existe de nombreuses bases de données d'arbres de gènes construits en utilisant diverses méthodes. Parmi celles-ci, on peut citer : Ensembl Compara [66], Hogenom[50], Phog[14], MetaPHOrs[53], PhylomeDB[28] et Panther[44]. Ces arbres de gènes contiennent souvent des erreurs qu'il faut corriger avant de les utiliser pour des analyses[25, 5, 55]. La liste des sources d'erreurs menant à des arbres de gènes erronés est diverse et variée. Parmi les causes d'erreur, on peut lister celles liées aux données et celles liées aux outils de reconstruction des arbres. Les erreurs liées aux données peuvent provenir de l'annotation des génomes, et des familles de gènes. D'autres sources d'erreurs peuvent être liées à la précision des algorithmes, car certains problèmes étant NP-complet, leur résolution se ramène à des solutions heuristiques ou des approximations qui ont des marges d'erreur plus ou moins grandes. À cela s'ajoutent les erreurs liées aux outils de regroupement de séquences et d'alignement de séquences.
Afin de corriger les arbres de gènes, des méthodes ont été développées. Étant donné un arbre G que l'on souhaite corriger, une première approche consiste à créer un espace d'arbres à explorer en effectuant des modifications topologiques aléatoires sur G. Parmi tous les arbres explorés, on retient le meilleur arbre suivant un critère donné. Cette approche est utilisée par exemple dans le programme Notung[8] pour corriger les arbres. On associe à chaque branche de l'arbre un support statistique. L'idée ici est de modifier la topologie au niveau des branches ayant de faibles supports statistiques. Une seconde approche pour corriger des erreurs dans les arbres de gènes consiste à contracter les branches à faible support pour ensuite les binariser adéquatement.

2.6 Conclusion

Ce second chapitre présente l'état de l'art des méthodes de reconstruction des arbres et met un accent particulier sur les arbres de gènes. Ce chapitre débute par

la présentation de l'utilité des phylogénies des gènes, puis aborde les grandes étapes généralement utilisées pour reconstruire ces phylogénies. Dans la seconde section, on présente progressivement les méthodes de reconstruction des arbres basées sur les distances, les méthodes de parcimonies, les méthodes de vraisemblance, les méthodes d'inférences bayésiennes, les méthodes de super-arbres et les méthodes basées sur la réconciliation. Par la suite, nous présentons les sources d'erreurs pouvant jouer sur la qualité des arbres, et nous présentons quelques approches pour corriger ces arbres. Le chapitre suivant va présenter plus en détail la reconstruction des arbres de gènes et d'arbres de transcrits en utilisant une nouvelle méthode : le regroupement de séquences. Le but du chapitre 3 suivant est de présenter au lecteur l'intérêt de dissocier les arbres de gènes des arbres de transcrits et les reconstruire séparément

Chapitre 3

Reconstruction de phylogénies de transcrits et de gènes en utilisant la réconciliation et le regroupement avec chevauchement

3.1 Introduction

Les études sur la génomique attestent que le mécanisme de l'épissage alternatif joue un rôle majeur dans la diversification des transcrits produits par les gènes chez les eucaryotes. On sait désormais que ces gènes ont la capacité de produire plus d'un transcrit. Plus spécifiquement chez l'humain, on estime que 95% des gènes utilisent ce mécanisme. Cette production multiple de transcrits chez les eucaryotes entraîne une diversité des fonctions biologiques réalisées par ces gènes. Ces connaissances sur la production multiple de transcrits par un gène viennent mettre fin au dogme qui affirmait qu'un gène ne produisait qu'un seul transcrit. De ce fait, de nombreuses méthodes doivent être révisées. Il s'agit notamment des méthodes de reconstruction d'arbres de gènes et de transcrits, d'alignement de transcrits, de regroupement de transcrits, etc.

Dans ce chapitre, nous proposons une nouvelle approche pour reconstruire les

arbres de gènes et de transcrits en tenant compte de l'épissage alternatif. Nous avons étendu le modèle de réconciliation entre un arbre de gènes et un arbre d'espèces à celui de la double réconciliation entre un arbre de transcrits, un arbre de gènes et un arbre d'espèces. Puis, nous introduisons deux nouveaux problèmes d'optimisation. Enfin, nous proposons deux algorithmes pour la construction des arbres de gènes et des arbres de transcrits suivant ce modèle.

Ce chapitre est structuré comme suit : nous commençons par présenter l'une des principales méthodes de reconstruction des arbres de gènes suivis de ses limites. Puis, nous présentons les deux principaux modèles utilisés pour la construction d'arbres de transcrits et leurs limites. Dans ce chapitre et les chapitres suivants, nous utiliserons les mots transcrits, protéine, coding sequence(CDS) pour désigner le même objet.

3.2 Méthode de construction d'arbres de gènes de Ensembl

Dans cette section, nous présentons la méthode de construction des arbres de gènes de la base de données Ensembl[66]. Ce choix est motivé par le fait que les arbres d'Ensembl sont les plus couramment utilisés, et la méthode de construction de ces arbres est représentative. Elle nous permettra d'illustrer les limites des méthodes existantes. La figure 3.1 présente les huit étapes du processus de construction des arbres de gènes.

La construction d'arbre de gènes de Ensembl débute en choisissant la plus longue protéine de chaque gène comme protéine de référence. Puis, le calcul d'un score de similarité entre chaque paire de protéines est effectué en utilisant BLAST[32]. Sur la base de cette similarité, un graphe est construit comme suit : les sommets sont les protéines, une arête existe entre une paire de protéines si leur similarité réciproque est supérieure à un seuil fixé. Par la suite, les composantes connexes sont extraites de ce graphe. Les séquences extraites de chaque composante connexe sont utilisées pour créer un arbre réconcilié par composante connexe. Ensuite, en utilisant les arbres réconciliés, les gènes orthologues et paralogues sont inférés.

Bien que les arbres d'Ensembl soient les plus utilisés, la méthode de construction

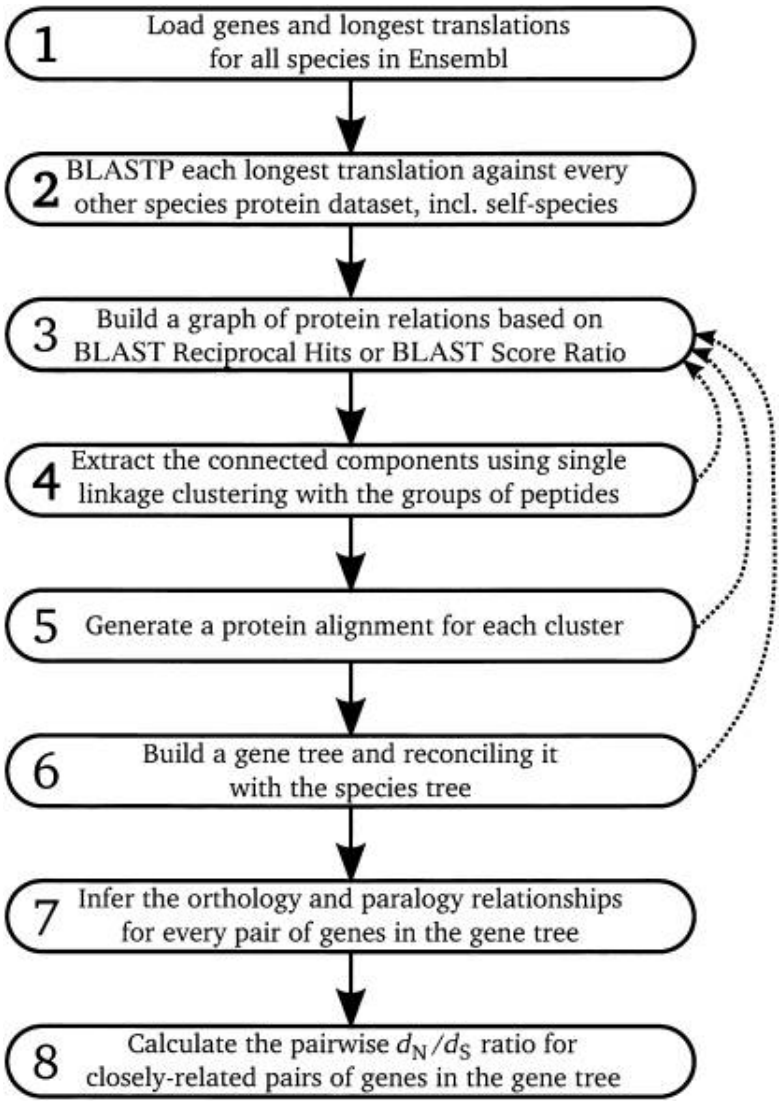


Figure 3.1 – Processus de construction d'arbre de gènes à huit étapes d'Ensembl.

de ces arbres présente un certain nombre de limitations que nous détaillons à la section suivante.

3.3 Limites de la méthode d'Ensembl

La méthodologie de construction d'arbres de gènes d'Ensembl présente d'importantes limites qui remettent en question la qualité des résultats obtenus. La principale limite est que cette méthode considère uniquement la plus longue protéine de chaque gène. Chaque gène étant donc représenté par sa plus longue protéine. Ainsi, l'arbre de gène obtenu n'est en réalité qu'un arbre des plus longues protéines de chaque gène.

La figure 3.2 présente un exemple de reconstruction de l'évolution d'un sous-ensemble de transcrits qui sera par la suite considéré comme arbre de gènes.

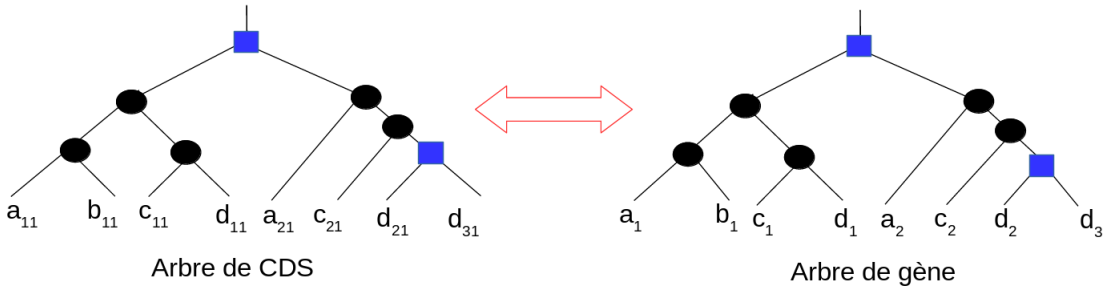


Figure 3.2 – Arbre de transcrits considéré comme arbre de gènes. Chaque gène est étiqueté par le symbole x_i et chaque CDS est étiqueté par le symbole x_{ij}. Le transcrit x_{ij} est produit par le gène x_i.

3.4 Méthode de construction d'arbres de gènes de IsoSel

Afin de pallier la principale limite des méthodes telle que Ensembl qui réside dans le choix du plus long transcrit par gène utilisé pour construire l'arbre de gènes, la méthode IsoSel [51] (*Isoform Selector*) a été proposée avec pour but la sélection de transcrits isoformes spécialement dédiés à la reconstruction d'arbres phylogénétiques.

La figure 3.3 présente les trois étapes de IsoSel. Étant donnée un ensemble de transcrits épissés, la première étape de IsoSel consiste à calculer un alignement multiple de ces transcrits. Cet alignement est considéré comme l'alignement de référence. La seconde étape consiste à générer un ensemble d'alignement en utilisant l'approche *bootstrap*. Chaque alignement est par la suite utilisé pour calculer une matrice de distance. Cette matrice de distance est utilisée pour construire un arbre guide servant à améliorer l'alignement initial. L'ensemble des alignements améliorés sont fusionnés afin d'obtenir un meilleur alignement multiple. La troisième et dernière étape de IsoSel consiste à calculer le score de la somme des paires (*Sum-of-Pairs score*) en utilisant l'alignement obtenu par l'approche *bootstrap*. Pour une séquence donnée, ce score représente la moyenne des scores de ces résidus. Pour chaque gène, le transcrit ayant le score de la somme des paires le plus élevé est utilisé pour faire partie du groupe de transcrits isoformes.

La principale limite de cette méthode vient du fait que le choix des transcrits isoformes est effectué sur la base de la comparaison des séquences de transcrits. Cette approche ne tient pas compte de la comparaison de la structure d'épissage entre transcrits.

3.5 Méthodes de construction d'arbres de transcrits

Contrairement à la notion d'arbres de gènes qui est bien connue, les méthodes de reconstruction d'arbres de transcrits sont relativement récentes[10, 11, 1]. Dans cette section nous présentons deux travaux sur la construction d'arbres de transcrits.

Christinat *et al.* ont présenté la problématique de la reconstruction de la phylogénie des transcrits[10]. Leur méthode est basée sur la structure des gènes et vise à reconstruire l'histoire évolutive des exons dans les gènes ancestraux afin d'en déduire l'évolution des transcrits.

Le modèle d'évolution a deux niveaux, celui des gènes et celui des transcrits. Du côté des gènes, l'évolution se matérialise par la variation de l'état des exons : constitutif, alternatif et absent. Un exon est dit constitutif s'il est présent dans tous

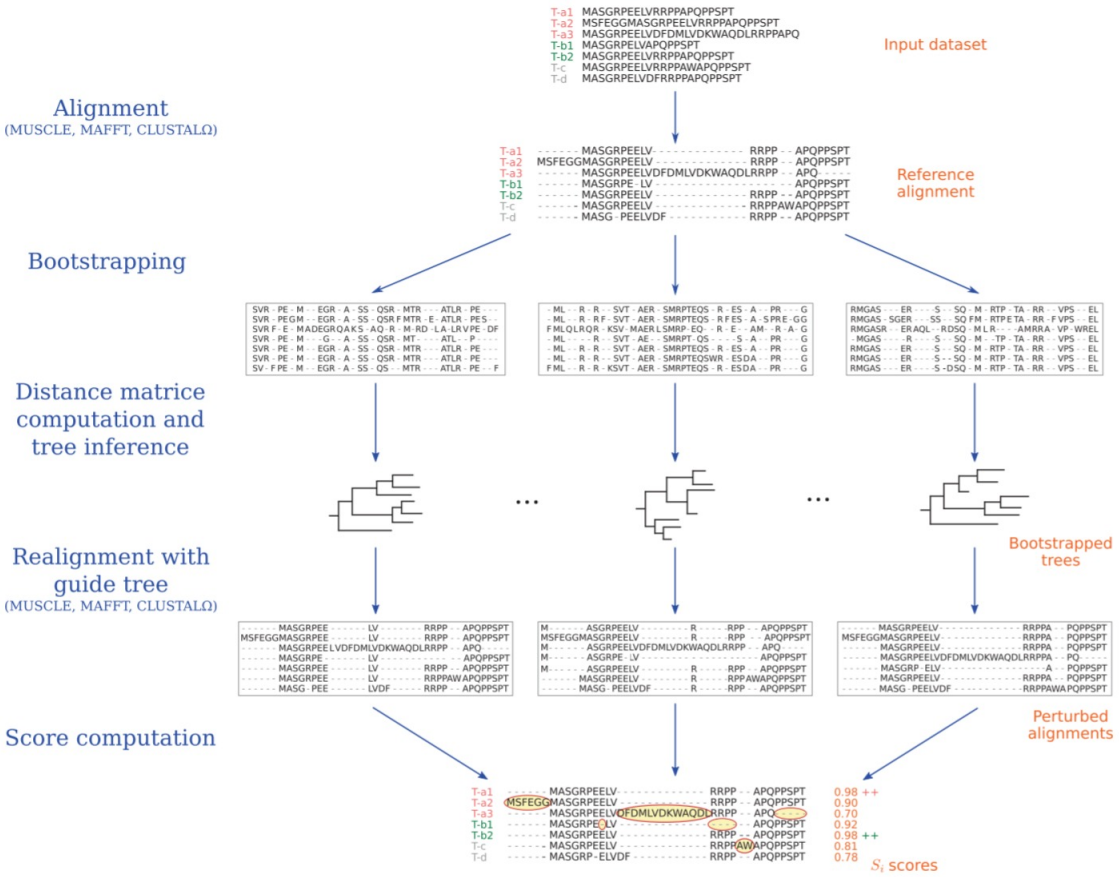


Figure 3.3 – Illustration des principales étapes pour la sélection de transcrits isoformes par IsoSel.

les transcrits produit par un gène, alternatif s'il est présent dans seulement certains transcrits produit par un gène et absent s'il l'est de tous les transcrits produit par un gène. Du côté des transcrits, l'évolution se matérialise par des événements de perte et de création de transcrits. Ces travaux ont exposé un modèle d'évolution pour les transcrits et le problème de la reconstruction d'arbres de transcrits. Cependant, la méthode de reconstruction présente les limites suivantes :

— Cette méthode utilise l'outil Mauve[13], qui est un système de construction d'alignements de génomes multiples en présence d'événements évolutifs à grande échelle tels que les réarrangements génomiques. Il n'est pas dédié aux transcrits alternatifs.

— Elle se base sur l'arbre de gènes reconnu comme erroné pour reconstruire l'arbre des transcrits.
— L'algorithme débute par la reconstruction d'une forêt d'arbres de transcrits. Chaque arbre de cette forêt représente l'évolution d'un ensemble de transcrits similaires. Par la suite, cette forêt d'arbres est fusionnée selon une approche heuristique chronophage.

Ait-hamlat *et al.* ont proposé une approche pour la reconstruction de l'histoire évolutive d'une famille de transcrits en tenant compte de la structure des transcrits[1]. Leur méthode se base sur un arbre de gènes comportant la liste des transcrits de chaque gène aux feuilles de l'arbre et cherche à reconstruire la forêt d'arbres qui explique mieux l'évolution des transcrits. La méthode est basée sur le principe du maximum de parcimonie. Cette méthode intègre la structure des transcrits, mais elle néglige la structure d'épissage des gènes, ce qui rend difficile la comparaison entre les transcrits de gènes différents. Les principales limites de cette approche sont :

— Qu'elle se base sur un arbre de gènes qui peut être erroné.
— Que la méthode de construction et de fusion de la forêt d'arbres est chronophage et donc peu adaptée en pratique.

De façon générale, on note une prise en compte partielle de l'information sur la structure des transcrits et l'absence d'algorithmes pour le regroupement des transcrits alternatifs.

3.6 Article : 'Reconstructing Protein and Gene Phylogenies using reconciliation and soft-clustering'

Afin de proposer des solutions aux limites des méthodes actuelles de reconstruction d'arbres de gènes et d'arbres de protéines, nous avons développé de nouveaux modèles et algorithmes pour la reconstruction d'arbres phylogénétiques basée sur la réconciliation. L'originalité de cette contribution repose sur le fait que nous différencions tout d'abord un arbre de gènes d'un arbre de protéines (transcrits). Puis, les arbres de protéines sont reconstruits en tenant compte de toutes les protéines d'une famille de gènes. Nous introduisons le concept de double réconciliation qui étend ce-

lui de la réconciliation entre deux arbres (un arbre de gènes et un arbre d'espèces) à trois arbres (un arbre de protéines, un arbre de gènes et un arbre d'espèces). En nous basant sur ce modèle, nous proposons une méthode pour le regroupement de séquences avec chevauchement pour regrouper les protéines. L'application de notre méthode pour la reconstruction d'arbres de gènes sur des données réelles a permis d'obtenir des arbres de meilleure qualité que ceux de la base de données d'Ensembl.

Pr. Ouangraoua a conçu l'étude. J'ai développé les algorithmes avec Pr. Ouangraoua. Pr. Lafond a produit les résultats sur la complexité des problèmes. J'ai développé le programme et rédigé sa documentation, collecté les données, mené les expériences et présenté les résultats lors de la conférence BICOB'2017. J'ai redigé le manuscrit avec Pr. Ouangraoua et Pr. Lafond. L'article étendu a été publié dans *journal Journal of Bioinformatics and Computational Biology* (JBCB).

Journal of Bioinformatics and Computational Biology

Reconstructing Protein and Gene Phylogenies using reconciliation and soft-clustering

Esaie Kuitche

Department of Computer Science, Université de Sherbrooke
Sherbrooke, QC J1K2R1, Canada
esaie.kuitche.kamela@USherbrooke.ca

Manuel Lafond

Department of Mathematics and Statistics, University of Ottawa
Ottawa, ON K1N6N5, Canada
mlafond2@UOttawa.ca

Aïda Ouangraoua

Department of Computer Science, Université de Sherbrooke
Sherbrooke, QC J1K2R1, Canada
aida.ouangraoua@USherbrooke.ca

Received (Day Month Year)
Revised (Day Month Year)
Accepted (Day Month Year)

The architecture of eukaryotic coding genes allows the production of several different protein isoforms by genes. Current gene phylogeny reconstruction methods make use of a single protein product per gene, ignoring information on alternative protein isoforms. These methods often lead to inaccurate gene tree reconstructions that require to be corrected before phylogenetic analyses. Here, we propose a new approach for the reconstruction of gene trees and protein trees accounting for alternative protein isoforms. We extend the concept of reconciliation to protein trees, and we define a new reconciliation problem called MinDRGT that consists in finding a gene tree that minimizes a double reconciliation cost with a given protein tree and a given species tree. We define a second problem called MinDRPGT that consists in finding a protein supertree and a gene tree minimizing a double reconciliation cost, given a species tree and a set of protein subtrees. We propose a shift from the traditional view of protein ortholog groups as hard-clusters to soft-clusters and we study the MinDRPGT problem under this assumption. We provide algorithmic exact and heuristic solutions for versions of the problems, and we present the results of applications on protein and gene trees from the Ensembl database. The implementations of the methods are available at https://github.com/UdeS-CoBIUS/Protein2GeneTree and https://github.com/UdeS-CoBIUS/SuperProteinTree.

Keywords: Protein Tree; Gene Tree ; Orthology; Reconciliation; Soft-clustering

1. Introduction

Recent genome analyses have revealed the ability of eukaryotic coding genes to produce several transcripts and proteins isoforms. This mechanism plays a major role in the functional diversification of genes [19,23]. Still, current gene phylogeny reconstruction methods make use of a single protein product per gene that is usually the longest protein called the "reference protein" ignoring the production of alternative protein isoforms [1,27,30]. It has been shown that these sequence-based methods often return incorrect gene trees [15,27]. Thus, several methods have been proposed for the correction of gene trees [24,31]. Recently, a few models and algorithms aimed at reconstructing the evolution of full sets of gene products along gene trees were introduced [7,32]. Some models have also been proposed to study the evolution of alternative splicing and gene exon-intron structures along gene trees [17,19]. All these models require the input of accurate gene trees and are biased when the input gene trees contain errors. Here, we explore a new approach in order to directly reconstruct accurate gene phylogenies and protein phylogenies while accounting for the production of alternative protein isoforms by genes. We introduce new models and algorithms for the reconstruction of gene phylogenies and full sets of proteins phylogenies using *reconciliation* [10] and *soft-clustering*. Reconciliation, first introduced by Goodman et al. in 1979 [11], is a widely used tool to explain the incongruence between a gene tree and a species tree (see e.g. [2,4,5,6,12,13,21,22] and [9] for a survey on reconciliation algorithms). Here, we also use reconciliation to compare protein trees and gene trees.

First, we present a model of protein evolution along a gene tree that involves two types of evolutionary events called *protein creation* and *protein loss*, in addition to the classical evolutionary events of speciation, gene duplication and gene loss considered in gene-species tree reconciliation. Second, we propose an extension of the framework of gene-species tree reconciliation in order to define the concept of protein-gene tree reconciliation, and we introduce new reconciliation problems aimed at reconstructing optimal gene trees and proteins trees. We define the problem of finding a gene tree minimizing the sum of the protein-gene and gene-species reconciliation costs, given the protein tree and the species tree. We call this problem the *Minimum Double Reconciliation Gene Tree* (MinDRGT) problem. We also define the problem of jointly finding a protein supertree and a gene tree minimizing the sum of the protein-gene and gene-species reconciliation costs, given the species tree and a set of subtrees of the protein tree to be found. We call this problem the *Minimum Double Reconciliation Protein and Gene Tree* (MinDRPGT) problem. Third, we define an adaptation of the UPGMA algorithm for the soft-clustering of proteins into ortholog groups and we use it for the reconstruction of protein supertrees.

The paper is organized as follows. We first formally define, in Section 3, the new protein evolutionary models and the related reconciliation problems, MinDRGT and MinDRPGT, for the reconstruction of gene phylogenies and full sets

of protein phylogenies. In Section 4, we prove the NP-hardness of some versions of MinDRGT, especially the one called MinDRGT$_{CD}$ that consists in minimizing the number of protein creation and gene duplication events. Next, in Section 5, we consider the MinDRGT problem in a special case where each gene is associated to a single protein. This restriction is relevant for the correction of gene trees output by sequence-based gene phylogeny reconstruction methods using a single protein per gene. Such methods make the unsupported assumption that each pair of leaf proteins in the protein tree is related through a least common ancestral node that corresponds to a speciation or a gene duplication event, and then, they output a gene tree equivalent to the reconstructed protein tree. In this perspective, the MinDRGT problem under the restriction that each gene is associated to a single protein, allows pairs of proteins to be related through ancestral protein creation events, and then asks to find an optimal gene tree, possibly different from the input protein tree. In other terms, the protein tree is not confused with the gene tree, but it is used, together with the species tree, to guide the reconstruction of the gene tree. We first show that, even with the restriction that each gene is associated to a single protein, for most versions of the MinDRGT problem, the optimal gene tree may differ from the input protein tree. In particular, we give a counterexample that shows that the duplication cost, the lost cost and the mutation cost considered in the classical gene-species tree reconciliation do not satisfy the triangle inequality. We then exhibit a heuristic algorithm called Protein2GeneTree for the MinDRGT problem that consists in building the optimal gene tree by applying modifications on the input protein tree guided by the species tree.

In Section 6, we consider the MinDRPGT problem aimed at jointly reconstructing both a protein phylogeny and a gene phylogeny. We consider a restriction on the input data that requires the set of input protein subtrees to be the set of all inclusion-wise maximum subtrees of the target protein supertree P that contain no protein creation node. Such an input consists of phylogenetic trees on clusters of orthologous proteins that can be obtained by using a soft-clustering approach to group proteins. Under this assumption, we present a polynomial-time exact algorithm called SuperProteinTree for computing the target protein supertree P, which allows to reduce MinDRPGT to a special case of MinDRGT where the input protein tree P is given with a partial labeling of its nodes. The algorithm consists in first reconstructing a partition of P composed of subtrees that are either entirely included or excluded in each input subtree, and then combining these partition subtrees into P.

In Section 7, the results of applying Protein2GeneTree for the correction of gene trees from the Ensembl database [16] show that the new framework allows to reconstruct gene trees whose double reconciliation costs are decreased, as compared to the initial Ensembl gene trees [30]. We also show that most (>80%) corrected trees cannot be rejected on a statistical basis as compared to the initial trees, according to the Approximately Unbiased (AU) Test [28]. Next, we present our modified UPGMA algorithm for protein soft-clustering and the results of applying it together with the

algorithm SuperProteinTree for the construction of protein supertrees. The results show that the reconstructed protein supertree supports the hypothesis of protein creation events that happened in ancestral genes with groups of orthologous protein isoforms shared by several extant genes.

2. Preliminaries: protein trees, gene trees and species trees

In this section, we introduce some preliminary notations: $\mathcal{S}$ denotes a set of species, $\mathcal{G}$ a set of genes representing a gene family, and $\mathcal{P}$ a set of proteins produced by the genes of the gene family. The three sets are accompanied with a mapping function $s : \mathcal{G} \to \mathcal{S}$ mapping each gene to its corresponding species, and a mapping function $g : \mathcal{P} \to \mathcal{G}$ mapping each protein to its corresponding gene. In the sequel, we assume that $\mathcal{S}$, $\mathcal{G}$ and $\mathcal{P}$ satisfy $\{s(x) : x \in \mathcal{G}\} = \mathcal{S}$ and $\{g(x) : x \in \mathcal{P}\} = \mathcal{G}$, without explicitly mentioning it.

Phylogenetic trees: A tree T for a set L is a rooted binary tree whose leafset is L. The leafset of a tree T is denoted by $\mathcal{L}(T)$ and the set of nodes of T is denoted by $\mathcal{V}(T)$. Given a node x of T, the complete subtree of T rooted at x is denoted by $T[x]$. The *lowest common ancestor* (lca) in T of a subset L' of $\mathcal{L}(T)$, denoted by $lca_T(L')$, is the ancestor common to all nodes in L' that is the most distant from the root of T. $T|_{L'}$ denotes the tree for L' obtained from $T[lca_T(L')]$ by removing every node that does not have a descendant in L', then contracting the nodes with a single child until none remains. Given an internal node x of T, the children of x are arbitrarily denoted by x_l and x_r.

Proteins, genes, and species trees: In the sequel, S denotes a species tree for the set $\mathcal{S}$, G denotes a gene tree for the set $\mathcal{G}$, and P denotes a protein tree for the set $\mathcal{P}$. The mapping function s is extended to be defined from $\mathcal{V}(G)$ to $\mathcal{V}(S)$ such that if x is an internal node of G, then $s(x) = lca_S(\{s(x') : x' \in \mathcal{L}(G[x])\})$, i.e. the image of a node $x \in \mathcal{V}(G)$ in $\mathcal{V}(S)$ is the lca in the tree S of all the images of the leaves of $G[x]$ by s. Similarly, the mapping function g is extended to be defined from $\mathcal{V}(P)$ to $\mathcal{V}(G)$ such that if x is an internal node of P, then $g(x) = lca_G(\{g(x') : x' \in \mathcal{L}(P[x])\})$.

Gene-species tree reconciliation: Each internal node of the species tree S represents an ancestral species at the moment of a speciation event (*Spec*) in the evolutionary history of $\mathcal{S}$. The gene tree G represents the evolutionary history of the genes of the gene family $\mathcal{G}$, and each internal node of G represents an ancestral gene at the moment of a *Spec* or a gene duplication event (*Dup*).

The *LCA-reconciliation* of G with S is a labeling function l_G from $\mathcal{V}(G)-\mathcal{L}(G)$ to $\{Spec, Dup\}$ such that the label of an internal node x of G is $l_G(x) = Spec$ if $s(x) \neq s(x_l)$ and $s(x) \neq s(x_r)$, and $l_G(x) = Dup$ otherwise (see e.g. [4,5,6,10,12,13,21,22]). The LCA-reconciliation induces gene loss events on edges of G as follows: given an edge (x, y) of the tree G such that $y = x_l$ or $y = x_r$, a gene loss event is induced on (x, y)

for each node located on the path between $s(x)$ and $s(y)$ in S (excluding $s(x)$ and $s(y)$). If $l_G(x) = Dup$ and $s(x) \neq s(y)$, an additional loss event preceding all other loss events is induced on (x, y) for $s(x)$. Figure 1 presents a gene tree G on a gene family $\mathcal{G} = \{a_2, a_3, b_0, b_1, b_2, b_3, c_1, c_2, c_3, d_3\}$ reconciled with a species tree S on a set of species $\mathcal{S} = \{a, b, c, d\}$.

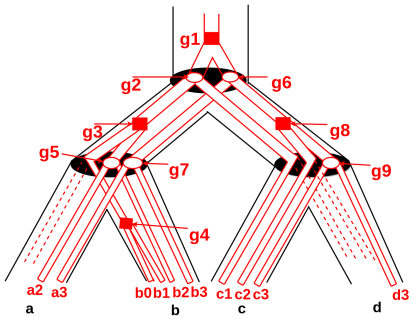


Fig. 1: A gene tree G on a gene family $\mathcal{G} = \{a_2, a_3, b_0, b_1, b_2, b_3, c_1, c_2, c_3, d_3\}$ such that $s(x_i) = x$ for any gene $x_i \in \mathcal{G}$ and species $x \in \mathcal{S} = \{a, b, c, d\}$. The species tree S is $((a, b), (c, d))$. G is reconciled with S: a speciation node x of G is located inside the node $l_G(x)$ of S, and a duplication node x is located on the edge $(p, l_G(x))$ of S such that p is the parent of $l_G(x)$ in S. The gene tree G contains 9 ancestral nodes: g_1, g_3, g_4, g_8 are duplications represented as squared nodes and g_2, g_5, g_6, g_7, g_9 are speciations represented as circular nodes. G contains 3 loss events whose locations are indicated with dashed edges. The same labeled gene tree G is represented in Figure 3 (Top), not embedded in S.

The LCA-reconciliation l_G suggests three possible costs of reconciliation $C_{G \to S}$ between G and S. The *duplication cost* denoted by $D(G, S)$ is the number of nodes x of G such that $l_G(x) = Dup$. The *loss cost* denoted by $L(G, S)$ is the overall number of loss events induced by l_G on edges of G. The *mutation cost* denoted by $M(G, S)$ is the sum of the duplication cost and the loss cost induced by l_G. In the example depicted in Figure 1, the duplication cost is 4, while the loss cost is 3, and the mutation cost is 7.

Homology relations between genes: Two genes x and y of the set $\mathcal{G}$ are called *orthologs* if $l_G(lca_G(\{x, y\})) = Spec$, and *paralogs* otherwise.

3. Model of protein evolution along a gene tree and problem statements

In this section, we first formally describe the new model of protein evolution along a gene tree. Next, we describe an extension of the framework of phylogenetic tree

reconciliation that makes use of the new model, and we state new optimization problems related to the extended framework.

Protein evolutionary model: The protein evolutionary model that we propose is based on the idea that the set of all proteins $\mathcal{P}$ produced by a gene family $\mathcal{G}$ have derived from a set $\mathcal{A}_P$ of common ancestral proteins that were produced by the ancestral gene located at the root of the gene tree G. This ancestral set of proteins evolved along the gene tree through different types of evolutionary and modification events including the classical events of speciation, gene duplication and gene loss. In the sequel, we consider that the ancestral set of proteins $\mathcal{A}_P$ is composed of a single ancestral protein that is the root of a tree for the set of proteins $\mathcal{P}$, but all definitions can be directly extended to protein forests, i.e sets of independent proteins trees rooted at multiple ancestral proteins.

A *protein tree* P is a tree for the set of proteins $\mathcal{P}$ representing the phylogeny of the proteins in $\mathcal{P}$. Each internal node of P represents an ancestral protein at the moment of a Spec, Dup, or a *protein creation event* (*Creat*). A protein creation event represents the appearance of a new protein isoform at a moment of the evolution of a gene family on an edge of the gene tree G. This evolutionary model is supported by recent studies on the evolution of gene alternative splicing patterns and inter-species comparison of gene exon-intron structures [17,19]. In particular, these studies have highlighted that alternative splicing patterns may be gene-specific or shared by groups of homologous genes [3,25]. A protein creation event thus leads to the observation of conserved protein isoforms called *orthologous splicing isoforms* [32] in a group of homologous extant genes descending from the ancestral gene that underwent the protein creation event. Based on these observations, the present model of protein evolution allows to describe the evolution of the full set of proteins produced by a gene family along the gene tree of the family. Figure 2 presents an example of labeled protein tree for a set of proteins $\mathcal{P} = \{a21, a31, b01, b02, b11, b21, b31, c11, c12, c21, c31, d31\}$.

Protein-gene tree reconciliation: We naturally extend the concept of reconciliation to protein trees as follows. The *LCA-reconciliation* of P with G is a labeling function l_P from $\mathcal{V}(P) - \mathcal{L}(P)$ to $\{Spec, Dup, Creat\}$ that labels an internal node x of P as $l_P(x) = Spec$ if $g(x) \neq g(x_l)$ and $g(x) \neq g(x_r)$ and $l_G(g(x)) = Spec$, else $l_P(x) = Dup$ if $g(x) \neq g(x_l)$ and $g(x) \neq g(x_r)$ and $l_G(g(x)) = Dup$, and $l_G(x) = Creat$ otherwise. Note that, if x is such that $\{g(y)|y \in \mathcal{L}(P[x_l])\} \cap \{g(y)|y \in \mathcal{L}(P[x_r])\} \neq \emptyset$, then $l_P(x) = Creat$, and x is called an *apparent creation* node.

Similarly to the LCA-reconciliation l_G, the LCA-reconciliation l_P induces protein loss events on edges of P as follows: given an edge (x, y) of P such that $y = x_l$ or $y = x_r$, a protein loss event is induced on (x, y) for each node located on the path between $g(x)$ and $g(y)$ in G. If $l_P(x) = Creat$ and $g(x) \neq g(y)$, an additional protein loss event preceding all other protein loss events is induced on (x, y) for $g(x)$. A protein loss event corresponds to the loss of the ability to produce a protein

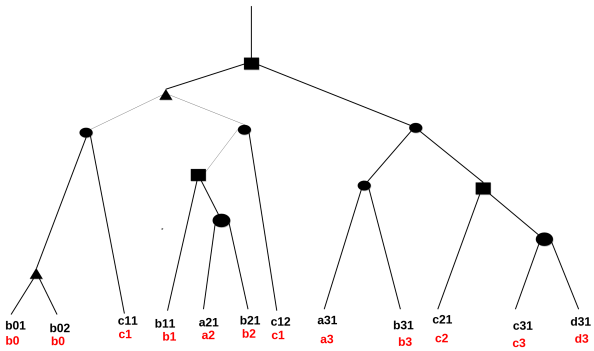


Fig. 2: A protein tree P on the set $\mathcal{P} = \{a21, a31, b01, b02, b11, b21, b31, c11, c12, c21, c31, d31\}$. The nodes of the tree are labeled as speciation (circular nodes), gene duplication (squared nodes), of protein creation events (triangular nodes). For each protein leaf x_{ij} of P, the corresponding gene $x_i = g(x_{ij})$ is indicated below the protein. The LCA-reconciliation that resulted in the labeling of the nodes of P is illustrated in Figure 3.

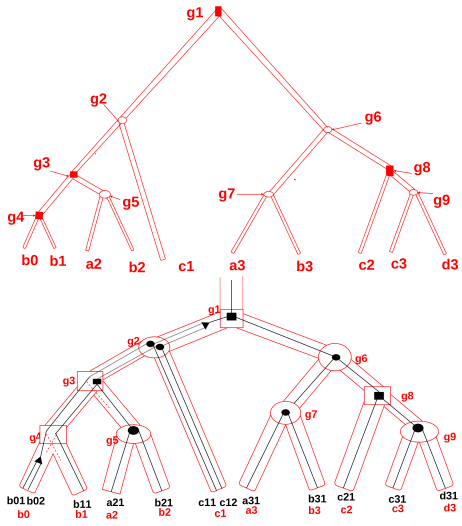


Fig. 3: Top. The labeled gene tree G of Figure 1. Bottom. The protein tree P of Figure 2 reconciled with G. For each internal node x of P, the corresponding image $g(x)$ in G is indicated. The protein tree P contains 2 protein creation nodes (triangular nodes), 3 gene duplication nodes (squared nodes), 6 speciation nodes (circular nodes), and 3 protein loss events indicated as dashed lines.

isoform for an ancestral gene at a moment of the evolution of the gene family.

We define the following three costs of reconciliation $C_{P\to G}$ induced by the LCA-reconciliation l_P of P with G. The *creation cost* denoted by $C(P,G)$ is the number of nodes x of P such that $l_P(x) = Creat$. The *loss cost* denoted by $L(P,G)$ is the overall number of loss events induced by l_P on edges of P. The *mutation cost* denoted by $M(P,G)$ is the sum of the creation cost and the loss cost induced by l_P. In the example depicted in Figure 3, the creation cost is 2, while the loss cost is 3, and the mutation cost is 5.

Homology relations between proteins: Based on the LCA-reconciliation of P with G, we can define the following homology relations between proteins of the set $\mathcal{P}$. Two proteins x and y of $\mathcal{P}$ are called *orthologs* if $l_P(lca_P(\{x,y\})) \neq Creat$, and in this case, we distinguish two types of orthology relationship: x and y are *ortho-orthologs* if $l_P(lca_P(\{x,y\})) = Spec$, and *para-orthologs* otherwise. Note that if x and y are ortho-orthologs (resp. para-orthologs), the genes $g(x)$ and $g(y)$ are orthologs (resp. paralogs). Finally, x and y are *paralogs* if $l_P(lca_P(\{x,y\})) = Creat$. Given a subset L' of $\mathcal{L}(P)$ such that any pair of proteins $(x,y) \in L' \times L'$ are orthologs, the tree $P|_{L'}$ induced by L' is called a *creation-free subtree* of P.

Problem statements: Given a protein tree P, a gene tree G and a species tree S, the *double reconciliation cost* of G with P and S is the sum of a cost $C_{P\to G}$ of reconciliation of P with G and a cost $C_{G\to S}$ of reconciliation of G with S.

Depending on the costs of reconciliation $C_{P\to G}$ considered for P with G, and $C_{G\to S}$ considered for G with S, nine types of double reconciliation cost can be defined. They are denoted by $XY(P,G,S)$ where X is either C for $C(P,G)$ or L for $L(P,G)$ or M for $M(P,G)$, and Y is either D for $D(G,S)$ or L for $L(G,S)$ or M for $M(G,S)$. For example, $CD(P,G,S)$ considers the creation cost for $C_{P\to G}$ and the duplication cost for $C_{G\to S}$.

The definition of the double reconciliation cost naturally leads to the definition of our first reconciliation problem that consists in finding an optimal gene tree G, given a protein tree P and a species tree S.

Minimum Double Reconciliation Gene Tree Problem (MinDRGT$_{XY}$):
Input: A species tree S for $\mathcal{S}$; a protein tree P for $\mathcal{P}$; a gene family $\mathcal{G}$.
Output: A gene tree G for $\mathcal{G}$ that minimizes the double reconciliation cost $XY(P,G,S)$.

The problem MinDRGT assumes that the protein tree P is known, but in practice, phylogenetic trees on full sets of proteins are not always available. Furthermore, the application of sequence-based phylogenetic reconstruction methods for constructing protein trees with more than one protein for some genes is likely to lead to incorrect trees, as for the reconstruction of single-protein-per-gene trees [15,27]. However, proteins subtrees of P can be obtained by building phylogenetic

trees for sets of orthologous protein isoforms [32]. Such subtrees can then be combined in order to obtain the full protein tree P. One way to combine the orthologous protein subtrees consists in following an approach, successfully used in [20] for combining a set of gene subtrees into a gene tree. It consists in jointly reconstructing the combined protein tree P and the gene tree G while seeking to minimize the double reconciliation cost of G with P and S. We then define a second problem that consists in finding an optimal pair of protein tree P and gene tree G, given a species tree S and a set of known subtrees $P_{i,1\leq i\leq k}$ of P.

Minimum Double Reconciliation Protein and Gene Tree Problem (MinDRPGT$_{XY}$):

Input: A species tree S for $\mathcal{S}$; a set of proteins $\mathcal{P}$, a set of subsets $\mathcal{P}_{i,1\leq i\leq k}$ of $\mathcal{P}$ such that $\bigcup_{i=1}^{k}\mathcal{P}_i=\mathcal{P}$, and a set of protein trees $P_{i,1\leq i\leq k}$ such that for each $i, 1\leq i\leq k$, P_i is a tree for $\mathcal{P}_i$ and a subtree of the target (real) protein tree.

Output: A protein tree P for $\mathcal{P}$ such that $\forall i$, $P|_{\mathcal{P}_i}=P_i$ and a gene tree G for $\mathcal{G}=\{g(x):x\in\mathcal{P}\}$ that minimize the double reconciliation cost $XY(P,G,S)$.

4. NP-hardness of MinDRGT

In this section, we prove the NP-hardness of MinDRGT$_{XY}$ for $X=C$ and $Y\in\{D,L,M\}$.

Proposition 1. *Given a protein tree P on $\mathcal{P}$ and a gene tree G on $\mathcal{G}$ with a protein-species mapping g, let G' be a gene tree on $\mathcal{G}'=\mathcal{P}$, and S' a species tree on $\mathcal{S}'=\mathcal{G}$ with a gene-species mapping $s=g$. The reconciliation costs from P to G, and from G' to S' satisfy the following: (1) $C(P,G)=D(G',S')$; (2) $L(P,G)=L(G',S')$; (3) $M(P,G)=M(G',S')$.*

From Proposition 1, all algorithmic results for the reconciliation problems between gene and species trees can be directly transferred to the equivalent reconciliation problems between protein and gene trees. In particular, in [22], it is shown that, given a gene tree G, finding a species tree S minimizing $D(G,S)$ is NP-hard. To our knowledge the complexity for the same problem with the $L(G,S)$ or $M(G,S)$ costs are still open, though we believe it is also NP-hard since they do not seem easier to handle than the duplication cost. Theorem 1 uses these results to imply the NP-hardness of some versions of MinDRGT.

Theorem 1. *Suppose that the problem of finding a species tree S minimizing the cost $Y'(G,S)$ with a given gene tree G is NP-hard for $Y'\in\{D,L,M\}$. Let $X=C$ if $Y'=D$, and $X=Y'$ if $Y'\in\{L,M\}$.*

Then for any reconciliation cost function $Y\in\{D,L,M\}$ and a given protein tree P and species tree S, the problem of finding a gene tree G minimizing the double-reconciliation cost $XY(P,G,S)$ is NP-hard, even if $|\mathcal{G}|=|\mathcal{S}|$.

Proof. Let $\{\hat{\mathcal{G}},\hat{G},\hat{\mathcal{S}},\hat{s}\}$ be an instance of the problem of finding an optimal species

tree minimizing Y', such that $\hat{\mathcal{G}}$ is the set of genes, $\hat{G}$ the gene tree, $\hat{\mathcal{S}}$ the set of species, and $\hat{s}$ is the gene-species mapping. We will reduce this problem to that of minimizing the double reconciliation cost for an instance $\{\mathcal{P}, P, \mathcal{G}, \mathcal{S}, S, g, s\}$. We create the protein set $\mathcal{P}$, protein tree P, species set $\mathcal{S}$ and species tree S with protein-gene mapping g and gene-species mapping s, as follows: $\mathcal{G} = \hat{\mathcal{S}}$ (i.e. the species become the genes), and $\mathcal{S}$ is such that $|\mathcal{S}| = |\mathcal{G}|$, and the mapping s a bijection between $\mathcal{S}$ and $\mathcal{G}$ (i.e. each species has one gene). The tree S is an arbitrary tree over leafset $\mathcal{S}$. Denote $n = |\mathcal{S}| = |\mathcal{G}|$. The protein set $\mathcal{P}$ consists in n^3 copies of $\hat{\mathcal{G}}$, i.e. $\mathcal{P} = \bigcup_{\hat{g}\in\hat{\mathcal{G}}}\{g_1, g_2, \ldots, g_{n^3}\}$. For each protein $g_i \in \mathcal{P}$ corresponding to gene $\hat{g} \in \hat{\mathcal{G}}$, we set the mapping $g(g_i) = \hat{s}(\hat{g})$. To construct the protein tree P, first create a set of n^3 copies $\mathbb{P} = \{P_1, \ldots, P_{n^3}\}$ of $\hat{G}$ such that for each $i \in [n^3]$, the tree P_i is obtained from $\hat{G}$ by replacing each gene $\hat{g} \in \mathcal{L}(\hat{G})$ by its i-th corresponding protein $g_i \in \mathcal{P}$. Next, let T be any binary tree with n^3 leaves $l_1, \ldots, l_{n^3}$, and for each $i \in [n^3]$, replace l_i by the tree P_i. Note that every of the $n^3 - 1$ internal node x initially in T must be an apparent creation node. Also note that no matter what the gene tree G is, x will be mapped to the root of G, as well as its two children x_l and x_r, implying that there are no losses on a branch incident to a node initially in T. The hardness of MinDRGT$_{XY}$ follows from the next (crude) upper bound:

Claim 1. *Let T_1, T_2 be two trees on the same leafset L. Then for any cost $Z \in \{D, L, M\}$, $Z(T_1, T_2) \leq 5|L|^2$.*

Proof. It suffices to prove the claim for the mutation cost M. When reconciled with T_2, the tree T_1 has at most $|\mathcal{V}(T_1)\setminus\mathcal{L}(T_1)| \leq |L|$ duplication nodes. As for the losses, each branch of T_1 induces at most $|\mathcal{V}(T_2)| \leq 2L$ losses, and so T_1 has a total of at most $|E(T_1)|2L \leq 2|L|2|L| = 4|L|^2$ losses. Thus $M(T_1, T_2) \leq 4|L|^2 + |L| \leq 5|L|^2$. □

We now show that there is a species tree $\hat{S}$ with $Y'(\hat{G}, \hat{S}) \leq k$ if and only if there is a gene tree G with double reconciliation cost $XY(P, G, S) \leq k(n^3 + 1) + 5n^2$.

($\Rightarrow$): let $\hat{S}$ be such that $Y'(\hat{G}, \hat{S}) \leq k$. Then our solution is $G = \hat{S}$. Since there are no losses on the branches incident to the nodes initially in T, and because each subtree P_i of $\mathbb{P}$ is a copy of $\hat{G}$, we have $X(P, G) = \sum_{P_i\in\mathbb{P}} X(P_i, G) + n^3 - 1 = \sum_{P_i\in\mathbb{P}} Y'(\hat{G}, \hat{S}) + n^3 - 1 \leq n^3(k + 1)$. Moreover, by Claim 1, $Y(G, S) \leq 5n^2$, and so the double reconciliation cost is at most $n^3(k + 1) + 5n^2$, as desired.

($\Leftarrow$): let G be a gene tree with $XY(P, G, S) \leq (k + 1)n^3 + 5n^2$. We claim that letting $\hat{S} = G$, we obtain a solution with $Y'(\hat{G}, \hat{S}) \leq k$. Suppose otherwise that $Y'(\hat{G}, \hat{S}) \geq k + 1$. Then as each P_i is a copy of $\hat{G}$ and $\hat{S} = G$, $\sum_{P_i\in\mathbb{P}} X(P_i, G) = \sum_{P_i\in\mathbb{P}} Y'(\hat{G}, \hat{S}) \geq (k + 1)n^3$. Moreover, there are $n^3 - 1$ creation nodes above the P_i's, and so $X(P, G) \geq (k + 2)n^3 - 1$. But for large enough n (i.e. $n \geq 6$), $(k+2)n^3-1 > (k+1)n^3+5n^2$, contradicting our assumption on $XY(P, G, S)$. This completes the proof.

Corollary 1. *The* MinDRGT$_{XY}$ *problem is NP-hard for $X = C$ and $Y \in \{D, L, M\}$.*

5. MinDRGT for the case $\mathcal{P} \Leftrightarrow \mathcal{G}$

In Section 4, we have proved the NP-hardness of several versions of the MinDRGT problem. In this section, we consider the problem in a special case where $\mathcal{P} \Leftrightarrow \mathcal{G}$, i.e each gene is the image of a single protein by the mapping function g. In the remaining of the section, we assume that $\mathcal{P} \Leftrightarrow \mathcal{G}$ without explicitly mentioning it. We first study the subcase where $\mathcal{P} \Leftrightarrow \mathcal{G} \Leftrightarrow \mathcal{S}$, i.e each species contains a single gene of the family. Next, we study the case where $\mathcal{P} \Leftrightarrow \mathcal{G}$ and develop a heuristic method for it. In the sequel, given a protein tree P on $\mathcal{P}$, $g(P)$ denotes the gene tree for $\mathcal{G}$ obtained from P by replacing each leaf protein $x \in \mathcal{P}$ by the gene $g(x)$ (see Figure 4 for example). Notice that for any cost function X, $X(P, g(P)) = 0$.

5.1. *Case where $\mathcal{P} \Leftrightarrow \mathcal{G} \Leftrightarrow \mathcal{S}$.*

In this section, we consider the additional restriction that $\mathcal{G} \Leftrightarrow \mathcal{S}$. For a gene tree G on $\mathcal{G}$, $s(G)$ denotes the species tree for $\mathcal{S}$ obtained from G by replacing each leaf gene $x \in \mathcal{G}$ by the species $s(x)$.

One question of interest is whether $g(P)$ is always a solution for MinDRGT$_{XY}$. In other words, is it the case that for any gene tree G', $Y(g(P), S) \leq X(P, G') + Y(G', S)$? When $X = C$ and $Y = D$, this is true if and only if the duplication cost satisfies the triangle inequality. In [22], the authors believed that the duplication cost did have this property, but as we show in Figure 4, this is not always the case. In fact, $g(P)$ cannot be assumed to be optimal also for the case $X = Y \in \{L, M\}$. Thus we get the following remark.

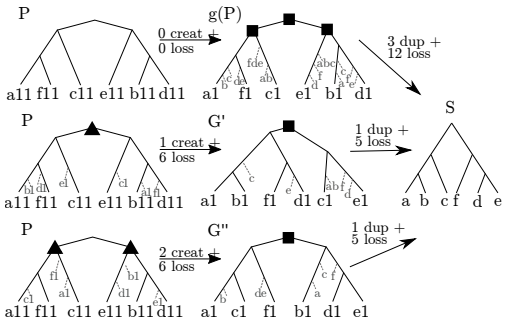


Fig. 4: Example of protein tree P on $\mathcal{P} = \{a_{11}, b_{11}, c_{11}, d_{11}, e_{11}, f_{11}\}$, species tree S on $\mathcal{S} = \{a, b, c, d, e, f\}$ and gene family $\mathcal{G} = \{a_1, b_1, c_1, d_1, e_1, f_1\}$, with $g(x_{ij}) = x_i$ and $s(x_i) = x$ for any protein $x_{ij} \in \mathcal{P}$, gene $x_i \in \mathcal{G}$ and species $x \in \mathcal{S}$. The gene tree G' on $\mathcal{G}$ induces a cost that is strictly lower than the cost induced by the gene tree $g(P)$, for the following double reconciliation costs : CD, CL, CM, LL, LM, MM. The gene tree G'' is the tree that Protein2GeneTree would output.

Remark 1. Figure 4 illustrates that under the restriction that $\mathcal{P} \Leftrightarrow \mathcal{G} \Leftrightarrow \mathcal{S}$, in particular the duplication cost, lost cost and the mutation cost do not satisfy the triangle inequality, i.e., there may exists a gene tree G' on $\mathcal{G}$ such that $D(g(P), S) > C(P, G') + D(G', S)$, $L(g(P), S) > L(g(P), s(G')) + L(G', S)$ and $M(g(P), S) > M(g(P), s(G')) + M(G', S)$. Moreover, the gene tree $g(P)$ is not a solution for any of MinDRGT$_{CD}$, MinDRGT$_{LL}$, MinDRGT$_{CL}$, MinDRGT$_{CM}$, MinDRGT$_{LL}$, MinDRGT$_{LM}$, and MinDRGT$_{MM}$.

5.2. *Case where $\mathcal{P} \Leftrightarrow \mathcal{G}$.*

In the following, we present a heuristic method for the MinDRGT$_{XY}$ problem, under the restriction that $|\mathcal{P}| = |\mathcal{G}|$. The intuition behind the algorithm is based on the idea that we seek for a gene tree G_{opt} on $\mathcal{G}$ that decreases the reconciliation cost with S, while slightly increasing the reconciliation cost with P, in order to globally decreases the double reconciliation cost with P and S. Let $G = g(P)$. The method consists in building G_{opt} from G by slightly modifying subtrees of G that are incongruent with S, in order to decrease the reconciliation cost with S. The heuristic method makes the following choices:
C1) the subtrees of G incongruent with S are those rooted at duplication nodes x with $l_G(x) \neq l_G(x_l)$ or $l_G(x) \neq l_G(x_r)$.
C2) If $l_G(x) = Dup$ and $l_G(x) \neq l_G(x_l)$ w.l.o.g, we denote by $Mix(G[x])$ the set of trees G' on $\mathcal{L}(G[x])$ that can be obtained by grafting $G[x_l]$ onto an edge of $G[x_r]$ on which $s(x_l)$ is lost. Then, the slight modification applied on $G[x]$ consists in replacing $G[x]$ by a tree $G' \in Mix(G[x])$ that decreases the double reconciliation cost with P and S by at least 1.

Protein2GeneTree: Heuristic for MinDRGT$_{XY}$
Input : Tree P on $\mathcal{P}$, Tree S on $\mathcal{S}$, gene set $\mathcal{G}$, mappings g, s.
Output : Tree G_{opt} on $\mathcal{G}$ such that $G_{opt} = g(P)$ or $XY(P, G_{opt}, S) < XY(P, g(P), S)$
1) $G \leftarrow g(P)$
2) Compute l_G and let $\mathcal{D} = \{x \in \mathcal{V}(G) \mid l_G(x) = Dup \text{ and } l_G(x) \neq l_G(x_l) \text{ or } l_G(x) \neq l_G(x_r)\}$
3) For any node $x \in \mathcal{D}$:
   a) $u \leftarrow$ the single node u of P s.t. $g(u) = x$;
   b) $v \leftarrow$ the single node v of S s.t. $v = s(x)$;
   c) $G_{opt}[x] \leftarrow argmax_{G' \in Mix(G[x])} XY(P[u], G', S[v])$
   d) $\delta(x) \leftarrow Y(G[x], S[v]) - XY(P[u], G_{opt}[x], S[v])$
4) Find a subset $\mathcal{D}'$ of $\mathcal{D}$ s.t. $\forall x \in \mathcal{D}'$, $\delta(x) > 0$, and $\forall (x, y) \in \mathcal{D}' \times \mathcal{D}'$, $lca(x, y) \neq x$ and $lca(x, y) \neq y$, and $\sum_{x \in \mathcal{D}'} \delta(x)$ is maximized.
5) Build G_{opt} from G by replacing any subtree $G[x], x \in \mathcal{D}'$ by $G_{opt}[x]$.

Complexity: For Step 4 of Protein2GeneTree, we use a linear-time heuristic greedy algorithm. The time complexity of Protein2GeneTree is in $O(n^2)$ where $n = |\mathcal{G}|$,

since $|\mathcal{V}(G)| = O(n)$, $|\mathcal{D}| = O(n)$ and $|Mix(G[x])| = O(n)$ for any $x \in \mathcal{D}$, and Steps 4 and 5 are realized in linear-time.

For example, the application of Protein2GeneTree on the example of protein tree P and species tree S depicted in Figure 4 would allow to reconstruct the gene tree G'' obtained by moving the subtree of $g(P)$ containing gene c_1 onto the branch leading to gene a_1, and moving the subtree containing gene e_1 onto the branch leading to gene d_1. However, the resulting gene tree G'' is not as optimal as the gene tree G'. Protein2GeneTree can be extended in order to allow computing the more optimal gene tree G' by modifying the choices C1 and C2 made by the algorithm: for example, in Step 2 set $\mathcal{D} = \{x \in V(G) \mid l_G(x) = Dup\}$, and in Step 3.c consider $Mix(G[x]) = \{G' \mid G'|_{\mathcal{L}(G[x_l])} = G[x_l] \text{ and } G|_{\mathcal{L}(G[x_r])} = G[x_r]\}$. The resulting algorithm would be an exponential time algorithm because of the exponential size of the sets $Mix(G[x])$.

6. MinDRPGT for maximum creation-free protein subtrees

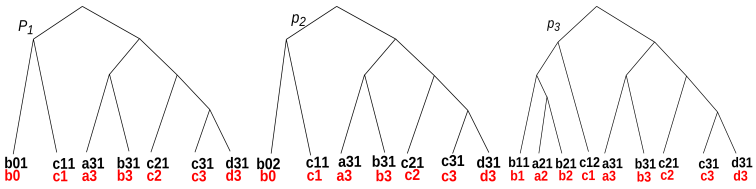
In this section, we consider the MinDRPGT problem in a special case where the input subtrees $P_{i,1\leq i\leq k}$ are all the inclusion-wise maximum creation-free protein subtrees of the real protein tree. For example, the inclusion-wise maximum creation-free protein subtrees of the labeled protein tree P depicted in Figure 2 are the subtrees P_1, P_2, P_3 of P induced by the subsets of proteins $\mathcal{P}_1 = \{b_{01}, c_{11}, a_{31}, b_{31}, c_{21}, c_{31}, d_{31}\}$, $\mathcal{P}_2 = \{b_{02}, c_{11}, a_{31}, b_{31}, c_{21}, c_{31}, d_{31}\}$, and $\mathcal{P}_3 = \{b_{11}, a_{21}, b_{21}, c_{12}, a_{31}, b_{31}, c_{21}, c_{31}, d_{31}\}$. We develop an exact algorithm for reconstructing the protein supertree P given the input subtrees $P_{i,1\leq i\leq k}$, which allows to reduce MinDRPGT to a special case of MinDRGT. The intuition behind the algorithm is to first reconstruct a partition of P into subtrees that are either entirely included or excluded in each input subtree P_i and then combine these partition subtrees into P. The following describes how the partition subtrees of P called *span partition* subtrees are inferred from the input maximum creation-free protein subtrees.

Let P be a protein tree for $\mathcal{P}$ with a LCA-reconciliation l_P, and $\mathbb{P} = \{P_1, P_2, \ldots, P_k\}$ the set of all the inclusion-wise maximum creation-free protein subtrees of P. We define the function *span* from the set of protein $\mathcal{P}$ to the set $2^{\mathbb{P}}$ of subsets of $\mathbb{P}$ such that, for any $x \in \mathcal{P}$, $span(x)$ is the subset of $\mathbb{P}$ such that x is a leaf of any tree in $span(x)$, and x is not a leaf of any tree in $\mathbb{P} - span(x)$. For example, for Figure 2, $span(b_{01}) = \{P_1\}$, $span(c_{11}) = \{P_1, P_2\}$, $span(b_{11}) = \{P_3\}$, $span(a_{31}) = \{P_1, P_2, P_3\}$.

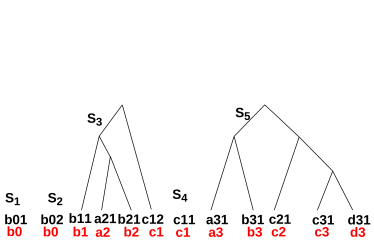
We define the *span partition* of $\mathcal{P}$ according to $\mathbb{P}$ as the partition $\mathbb{P}_{span} = \{S_1, S_2, \ldots, S_m\}$ of $\mathcal{P}$ such that for any set $S_u \in \mathbb{P}_{span}$, for any pair of proteins x, y in S_u, $span(x) = span(y)$. Note that $\mathbb{P}_{span}$ is unique. The function *span* is extended to be defined from $\mathcal{P} \cup \mathbb{P}_{span}$ to $2^{\mathbb{P}}$ such that for $S_u \in \mathbb{P}_{span}$, $span(S_u) = span(x)$ for any $x \in S_u$. The function g is extended to be defined from $\mathcal{P} \cup \mathbb{P}_{span}$ to $2^{\mathcal{G}}$ such that for $S_u \in \mathbb{P}_{span}$, $g(S_u)$ is the set of gene of proteins that belong to S_u.

For example, Figure 5 gives an illustration of the construction of the set of span

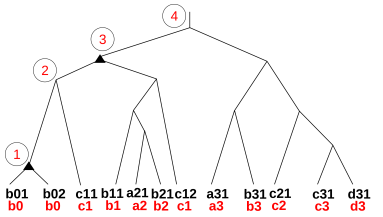
partition subtrees from an input set of maximum creation-free protein subtrees. Say we have the input set of maximum creation-free protein subtrees $\mathbb{P} = \{P_1, P_2, P_3\}$ (Figure 5a). The span partition of $\mathcal{P}$ according to $\mathbb{P}$ is $\mathbb{P}_{span} = \{S_1 = \{b_{01}\}, S_2 = \{b_{02}\}, S_3 = \{b_{11}, a_{21}, b_{21}, c_{12}\}, S_4 = \{c_{11}\}, S_5 = \{a_{31}, b_{31}, c_{21}, c_{31}, d_{31}\}\}$, and $span(S_1) = \{P_1\}$, $span(S_2) = \{P_2\}$, $span(S_3) = \{P_3\}$, $span(S_4) = \{P_1, P_2\}$ and $span(S_5) = \{P_1, P_2, P_3\}$. The set of span partition subtrees is then obtained by building a tree for each element of the span partition (Figure 5b).



(a) Input set of maximum creation-free protein subtrees $\mathbb{P} = \{P_1, P_2, P_3\}$



(b) Span partition trees



(c) Output partially labeled protein supertree

Fig. 5: Illustration of the steps of the algorithm ProteinSuperTree on (a) an input set of maximum creation-free protein subtrees $\mathbb{P} = \{P_1, P_2, P_3\}$. (b) The span partition of $\mathcal{P}$ according to $\mathbb{P}$ contains 5 sets. (c) The output supertree P with numbers in circles representing the order of span partition subtrees merging in the algorithm.

The following lemma describes a property of a set of span partition subtrees that allows to reconstruct a super protein tree from its set of span partition subtrees.

Lemma 1. *Let $\mathbb{P}$ be the set of all maximum creation-free protein subtrees of a labeled protein tree P and $\mathbb{P}_{span}$ be the span partition of $\mathcal{P}$ according to $\mathbb{P}$.*

If P contains at least one protein creation node, then there exist at least a pair of distinct sets S_u, S_v in $\mathbb{P}_{span}$ such that the subtrees of P induced by S_u and S_v, $P|_{S_u}$ and $P|_{S_v}$, are complete subtrees of P, and the subtree of P induced by $S_u \cup S_v$ is also a complete subtree of P. In this case:
(1) $l_P(lca_P(S_u \cup S_v)) = Creat$;
(2) For any $t \in \{u, v\}$ and any $P_i \in span(S_t)$, $P|_{S_t} = P_i|_{S_t}$;
(3) $span(S_u) \cap span(S_v) = \emptyset$;

(4) the following two sets of subtrees are equal: $\{P_i|_{\mathcal{L}(P_i)-S_u} \mid P_i \in span(S_u)\} = \{P_i|_{\mathcal{L}(P_i)-S_v} \mid P_i \in span(S_v)\}$.

Proof. There must exist a node w in P such that $l_P(w) = Creat$ and no node $x \neq w$ in $P[w]$ satisfies $l_P(x) = Creat$. Then, $S_u = \mathcal{L}(P[w_l])$ and $S_v = \mathcal{L}(P[w_r])$. □

For any set $S_t \in \mathbb{P}_{span}$, $tree(S_t)$ denotes the (possibly partial) subtree of P such that, for any $P_i \in span(S_t)$, $tree(S_t) = P_i|_{S_t}$.

SuperProteinTree: for reconstructing P from the input set $\mathbb{P}$
Input : Set $\mathbb{P}$ of all inclusion-wise maximum creation-free protein subtrees of a target protein tree P on $\mathcal{P}$, gene set $\mathcal{G}$, mappings g
Output : Protein tree P.
1) Compute the span partition, $Q \leftarrow \mathbb{P}_{span} = \{S_1, \ldots, S_m\}$;
2) Set $\{tree(S_u) \mid S_u \in \mathbb{P}_{span}\}$ as subtrees of P ;
3) While $|Q| > 1$:
a) Compute the set C of pairs of distinct elements (S_u, S_v) of Q^2 such that $span(S_u) \cap span(S_v) = \emptyset$ and $\{P_i|_{\mathcal{L}(P_i)-S_u} \mid P_i \in span(S_u)\} = \{P_i|_{\mathcal{L}(P_i)-S_v} \mid P_i \in span(S_v)\}$.
b) If $C \neq \emptyset$:
i) Pick a pair (S_u, S_v) in C such that S_u and S_v have the highest mean similarity score between their protein sets;
ii) Add a node w in P such that $tree(S_u)$, $tree(S_v)$ become the left and right subtrees of w, resulting in a subtree P';
iii) Set $l_P(w) \leftarrow Creat$ and for any $S_t \in Q$, $span(S_t) \leftarrow span(S_t) - span(S_v)$;
c) Otherwise,
i) Compute the set of pair of distinct elements (S_u, S_v) of Q^2 such that $P|_{S_u}$ was built at a previous iteration of Step 3 and $span(S_u) = span(S_v)$ and $g(S_u) \cap g(S_v) = \emptyset$;
ii) Pick the pair (S_u, S_v) such that S_u and S_v have the highest mean similarity score between their protein sets;
iii) Graft $P|_{S_u}$ onto $P|_{S_v}$ as the sibling of the node of $P|_{S_v}$ such that the resulting tree P' on $S_u \cup S_v$ is compatible with all subtrees $P_i \in span(S_v)$, i.e. $P'|_{S_t} = P_i|_{S_t}$ with $S_t = (S_u \cup S_v) \cap \mathcal{L}(P_i)$;
d) Set $S_w \leftarrow S_u \cup S_v$ and $Q \leftarrow Q - \{S_u, S_v\} \cup \{S_w\}$ with $span(S_w) \leftarrow span(S_u)$;

Steps 1) and 2) of the algorithm that consist in the computation of the span partition subtrees are illustrated in Figures 5a and 5b . Step 3) that iteratively merges the span partition subtrees in order to obtain the partially labeled protein supertree P is depicted in Figure 5c.

Theorem 2. *Given the set $\mathbb{P}$ of all inclusion-wise maximum creation-free protein subtrees of a labeled protein tree P on $\mathcal{P}$, SuperProteinTree reconstructs P and its time complexity is in $O(n^3)$.*

Proof. The proof follows from the two points below:
(1) For any $S_u \in \mathbb{P}_{span}$, $tree(S_u)$ is a (possibly partial) subtree of P ;
(2) At the end of each iteration of Loop 3, ($Invariant_1$) the subtree P' constructed is a complete subtree of P, and ($Invariant_2$) for any set S_u in Q, there is at most one set $S_v \neq S_u$ in Q such that $span(S_u) = span(S_v)$:

(i) At the first iteration, Q necessarily satisfies case (3.b), and by Lemma 1, the subtree $P' = P|_{S_u \cup S_v}$ must be a complete subtree of P. Moreover, at the end of the iteration, $Invariant_2$ is satisfied.

(ii) Now, let $k \geq 2$. We show that if at the end of each of the first $k-1$ iterations of Loop 3, $Invariant_1$ and $Invariant_2$ were satisfied, then, they are also satisfied at the end of the k^{th} iteration. At the k^{th} iteration:

• If we are in case (3.b) of the algorithm, then there exist two distinct sets S_u, S_v in Q such that $span(S_u) \cap span(S_v) = \emptyset$ and $\{P_i|_{\mathcal{L}(P_i)-S_u} \mid P_i \in span(S_u)\}$ $= \{P_i|_{\mathcal{L}(P_i)-S_v} \mid P_i \in span(S_v)\}$. If S_u and S_v were both elements of the initial partition $Q = \mathbb{P}_{span}$, then from Lemma 1, $P' = P|_{S_u \cup S_v}$ is a complete subtree of P . If one of them, say S_u was not an element of $\mathbb{P}_{span}$, then $P|_{S_u}$ is a complete subtree of P by hypothesis.

• If we are in case (3.c) i.e $span(S_u) = span(S_v)$, then S_u and S_v can not be both elements of $\mathbb{P}_{span}$. So, by hypothesis, one of them, say S_u, is such that $P|_{S_u}$ is a complete subtree.

• So, in both cases (3.b) and (3.c), either $P' = P|_{S_u \cup S_v}$ is a complete subtree of P, or one of the two merged subtrees, say $P|_{S_u}$, is a complete subtree of P. In the later case, suppose that $P|_{S_u \cup S_v}$ is not a complete subtree of P. Then, there exists a protein $x \in \mathcal{P} - \{S_u \cup S_v\}$ such that $x \in \mathcal{L}(P|_{S_v})$ but $x \notin \mathcal{L}(P|_{S_u})$. Let $S_t \in Q$ be the set such that $x \in S_t$. We have $span(S_u) \subseteq span(S_t) \subseteq span(S_v)$, and then $span(S_u) = span(S_t) = span(S_v)$. So, at the end of the $(k-1)^{th}$ iteration, we had $span(S_t) = span(S_u) = span(S_v)$, which is impossible.

• At the end of the k^{th} iteration, $Invariant_2$ is satisfied, otherwise there were two sets S_t, S_q in Q at the end of the $(k-1)^{th}$ iteration such that $span(S_t) = span(S_q)$, which is impossible.

The time complexity of SuperProteinTree is in $O(n^3)$ since the loop at Step 3 is in $O(n)$ and Steps 3.a, 3.b and 3.c are in $O(n^2)$. □

Applying the algorithm SuperProteinTree on an instance of MinDRPGT such that the input subtrees $P_{i,1 \leq i \leq k}$ are all the inclusion-wise maximum creation-free protein subtrees of the real protein tree P, allows to reconstruct P with a partial labeling l_P indicating all protein creation nodes. Then, MinDRPGT is reduced to MinDRGT in the special case where a partial labeling of the input protein tree is given.

7. Application

7.1. *Protein2GeneTree*

We applied the algorithm Protein2GeneTree for the reconstruction of gene trees using protein trees and gene families of the Ensembl database release 87 [16]. Some of the trees were left unchanged by the algorithm. We call an Ensembl gene tree G *modified* if Protein2GeneTree, when given G, outputs a different tree. Otherwise we say that G is *unmodified*. The results are summarized in Table 1. They show that initial gene trees, and particularly large size trees, are predominantly suboptimal in terms of double reconciliation cost. Moreover, modified and unmodified trees have comparable numbers of duplications, but modified trees have significantly higher number of losses, suggesting that gene trees with many losses are susceptible to correction.

Table 1: Results of Protein2GeneTree on 12680 Ensembl gene trees. Samples: (A) $1 \leq n \leq 9$ (7500 trees), (B) $10 \leq n \leq 99$ (4386 trees), (C) $100 \leq n \leq 199$ (773 trees), where n is the number of leaves in a tree with the number of trees in each sample in parenthesis (1) Number and percentage of modified trees, (2) Average number of duplications / losses in unmodified trees, (3) Average number of duplications / losses in modified trees (before modification), (4) Average value / percentage of double reconciliation cost reduction on modified trees, (5) Average running time in ms, (6) Percentage of Ensembl trees that pass the AU test / percentage of corrected trees that pass the AU test, with the percentage of corrected trees with a better AU value in parentheses.

	(1)	(2)	(3)	(4)	(5)	(6)
(A)	72 / 0.96%	0.88/ 7.97	1.61/ 36.61	10.18/ 19.69%	15	93.2% / 71.2% (39.0%)
(B)	1734/ 39.53%	3.83/ 25.04	11.26/ 114.95	7.17/ 5.82%	328	80.6% / 81.1% (45.9%)
(C)	505/ 65.32%	31.54/ 168.8	31.38/ 310.62	16.42/ 4.42%	4685	81.7% / 80.4% (51.1%)

In order to demonstrate the relevance of the gene tree correction achieved using Protein2GeneTree, the corrected trees were evaluated using the AU (Approximately Unbiased) test [28]. This test compares the likelihood of a set of trees in order to build a confidence set, which consists of the trees that cannot be statistically rejected as a valid hypothesis. We say a tree can be rejected if its AU value, which is interpreted as a p-value, is under 0.05. Otherwise, no significant evidence allows us to reject one of the two trees. We start by executing PhyML [14] on the trees to obtain the log-likelihood values per site and we execute Consel [29] to obtain the results from the AU test.

Column 6 of Table 1 shows the percentage of the Ensembl trees/corrected trees that could not be rejected. The corrected trees were in the confidence set about as often as Ensembl, with the exception of the trees with less than 10 leaves. In total, 80.7% of the corrected trees were part of the confidence set and could not be statistically rejected. As for the Ensembl gene trees, 81.2% of them were part of the confidence set. Therefore, despite the fact that our correction algorithm does not consider the likelihood, the majority of the corrected trees represent a hypothesis that cannot be rejected on a statistical basis. Moreover, the corrections are significantly better than the original trees as often as the originals are better than the correction. For the scores themselves, in total there were 52.8% of the Ensembl gene tree that obtained a better AU score than the corrected trees, versus 46.8% of the corrected trees having a better AU score (the remaining 0.4% of the trees having equal value).

7.2. *SuperProteinTree*

We applied the algorithm SuperProteinTree on two sets of homologous genes from the Ensembl-Compara database release 89 [8] augmented with simulated data. For each gene family, we wanted to reconstruct a putative set $\mathbb{P}$ of inclusion-wise maximum creation-free protein subtrees, and build the protein supertree P from $\mathbb{P}$.

Dataset: The dataset contains 14 genes with their CDS sequences from two gene families, FAM86 and TP63, 7 genes per family, 25 CDS for FAM86 and 72 for TP63. For each gene family, the genes are from six different amniote species which are *human, chimpanzee, mouse, rat, cow and chicken*, including two paralogous genes for *human*. For each gene family, the set of CDS of each gene is augmented with simulated CDS as follows. Given a gene G and any CDS C from another gene in the same gene family, the spliced alignment of the CDS C on the gene G is computed using the spliced alignment tool SPlign [18]. If all exons composing C are aligned onto G, the regions of G aligned with exons of C are excised and joined together to simulate an alternative splicing resulting in a CDS C_G of G that is orthologous to the CDS C. If the CDS C_G is not already an existing CDS of the gene G, it is added to its set of CDS as a simulated CDS. By applying this procedure on each pair of CDS and gene in a gene family, we obtain an augmented dataset composed of the same 7 genes per family with their initial sets of CDS augmented with additional CDS, simulated using existing CDS of other genes. Table 2 gives more details about the dataset.

Moreover, the augmented dataset of CDS is partitioned into a set of hard clusters representing protein orthology groups based on the orthology relations inferred using the SPlign spliced alignments. For the the gene family FAM86, 8 orthology groups were inferred, numbered from Group 0 to Group 7, and for the gene family TP63, 24 groups were inferred, numbered from Group 0 to Group 23. The name of each CDS in Figures 6 and 7 is prefixed with the putative orthology group and the

gene to which it belongs.

Table 2: Detailed description of the dataset for the application of the algorithm ProteinSuperTree.

	Family			
	FAM86		**TP63**	
Species	**Gene ID**	**#CDS**	**Gene ID**	**#CDS**
Human	ENSG00000158483	3 ; 0	ENSG00000073282	11 ; 3
Human	ENSG00000186523	4 ; 3	ENSG00000078900	9 ; 0
Chimpanzee	ENSPTRG00000007738	1 ; 1	ENSPTRG00000015733	1 ; 9
Mouse	ENSMUSG00000022544	1 ; 2	ENSMUSG00000022510	8 ; 2
Rat	ENSRNOG00000002876	1 ; 3	ENSRNOG00000001924	5 ; 4
Cow	ENSBTAG00000008222	1 ; 3	ENSBTAG00000015460	1 ; 9
Chiken	ENSGALG00000002044	1 ; 1	ENSGALG00000007324	2 ; 8
Total		12 ; 13		37 ; 35

For each gene family, the family identifier is given together with, for each gene, the species, the Ensembl identifier, the number of CDS (initial CDS ; simulated CDS) of the genes.

Soft-clustering and construction of a putative set of maximum creation-free protein subtrees: Before applying SuperProteinTree, the set $\mathbb{P}$ of all inclusion-wise maximum creation-free protein subtrees of P should be computed. As the target tree P is not given as input, we designed a heuristic method for computing the putative set of subtrees $\mathbb{P}$. The first step of the method consists in soft-clustering the proteins into ortholog groups. The second step consists in building a subtree for each of the ortholog groups.

The soft-clustering step uses a distance measure d that associates to a pair of protein $(x, y) \in \mathcal{P}$ a distance $d(x, y)$ that is obtained from a linear combination of the pairwise similarity score of their nucleotide sequence alignment and a structural similarity score. The multiple alignment of all the CDS of a gene family is computed using the coding sequence alignment tool MACSE [26], and all pairwise alignments between proteins are extracted from the multiple alignment. The structural similarity is evaluated as the number of pairs of exons in the two proteins that are aligned together divided by the maximum number of exons in the two proteins. For each gene family, we used different pairs of values (α,β) for the weights associated to the structural similarity and the sequence similarity in the linear combination for the definition of the distance measure d ((1.0,0.0), (0.8,0.2), (0.6,0.4), (0.4,0.6), (0.2,0.8), and (0.0,1.0)). For each pair of weights for the linear combination, the distance measure d was normalized to a range between 0 and 1.

For the soft-clustering step, the first aim is to allow a protein to appear

in more than one cluster. For example, for the protein tree depicted in Figure 2, the target ortholog soft-clusters are the maximum creation-free subtrees $\mathcal{P}_1 = \{b_{01}, c_{11}, a_{31}, b_{31}, c_{21}, c_{31}, d_{31}\}$, $\mathcal{P}_2 = \{b_{02}, c_{11}, a_{31}, b_{31}, c_{21}, c_{31}, d_{31}\}$, and $\mathcal{P}_3 = \{b_{11}, a_{21}, b_{21}, c_{12}, a_{31}, b_{31}, c_{21}, c_{31}, d_{31}\}$. The second aim of the soft-clustering is to prevent a cluster to contain more than one protein from each gene, as in such a case two proteins of a same gene in a cluster would be paralogs.

Our clustering algorithm is the UPGMA (Unweighted Pair Group Method with Arithmetic Mean) hierarchical clustering algorithm modified as follows. (1) When a new cluster K resulting from the merging of two minimum-distance clusters I and J is added, the clusters I and J are not discarded. They remain candidates for future merging with other clusters. (2) Two clusters containing proteins from the same gene cannot be merged. (3) At each step, the distance between a new cluster K and a cluster L is computed as $D(K, L) = \max\{d(x, y) \mid (x, y) \in K \times L\}$.

For each gene family and for each pairs of values (α,β) for the weights of the structural similarity and the sequence similarity in the definition of the distance measure d, we applied the soft-clustering algorithm and outputed different sets of soft clusters obtained obtained after different numbers γ of iterations in the clustering ($n/2, n, 2n$ and $3n$ where n is the number of proteins in $\mathcal{P}$). Each value of the number of iterations γ in the clustering then resulted in a set of soft clusters for the set of protein P. For a given set of soft clusters obtained for a set of parameters (α,β, γ), the corresponding set of input subtrees for the SuperProteinTree algorithm was computed as follows. PhyML [14] was used to build an initial tree P' for the set of all proteins P. For each ortholog group K in the set of soft clusters, the subtree of P for K was then defined as the subtree of P' induced by K.

Results of applying the SuperProteinTree algorithm: For each gene family and each set of input subtrees obtained for different values of the parameters α, β, and γ in the hierarchical soft clustering, we applied the algorithm SuperProteinTree to reconstruct a protein supertree. We then retained the best reconstructed protein supertree, i.e one that consists in a single tree (not a forest composed of incompatible subtrees) with the minimum number of creation events.

FAM86: Table 3 summarizes the results for the gene family FAM86. Figure 6 shows the initial protein tree P' computed using PhyML [14] and the protein supertree P reconstructed using SuperProteinTree for the best combination of parameter values (given in Table 3), $\alpha = 0.8$, $\beta = 0.2$ and $\gamma = n = 25$. The tree P' contains 18 proteins creation events while the supertree P contains 8 proteins creation events.

TP63: Table 4 summarizes the results for the gene family TP63. Figure 7 shows the initial protein tree P' computed using PhyML and the protein supertree P reconstructed using SuperProteinTree for the best combination of parameter values, $\alpha = 0.8$, $\beta = 0.2$, and $\gamma = n = 72$. The tree P' contains 62 proteins creation events while the supertree P contains 28 proteins creation events.

Table 3: SuperProteinTree results obtained on $n = 25$ proteins for the gene family FAM86 for varying values of the parameters α (weight of the structure similarity in the distance measure d), β (weight of the sequence similarity), and γ (number of iteration of the hierarchical soft clustering). For each of set of values, three results **A, B, C** are given: **A** is a binary value that indicates if SuperProteinTree has reconstructed a single protein tree (1) or a forest (0); **B** is the number of creation events inferred in the reconstruction and **C** is the ratio of the number of recovered initial orthology groups (8 initial orthology groups for FAM86). The best results are indicated in bold font.

		γ											
		n/2			**n**			**2n**			**3n**		
α	β	**A**	**B**	**C**	**A**	**B**	**C**	**A**	**B**	**C**	**A**	**B**	**C**
1.0	**0.0**	1	17	1/8	1	18	2/8	0	12	2/8	0	7	1/8
0.8	**0.2**	1	15	3/8	**1**	**8**	**6/8**	0	10	6/8	0	9	3/8
0.6	**0.4**	1	17	1/8	1	16	1/8	0	9	3/8	0	8	3/8
0.4	**0.6**	1	17	1/8	0	15	1/8	0	12	5/8	0	6	1/8
0.2	**0.8**	0	16	1/8	1	17	1/8	0	13	3/8	0	7	1/8
0.0	**1.0**	1	18	1/8	0	15	1/8	1	16	2/8	0	14	1/8

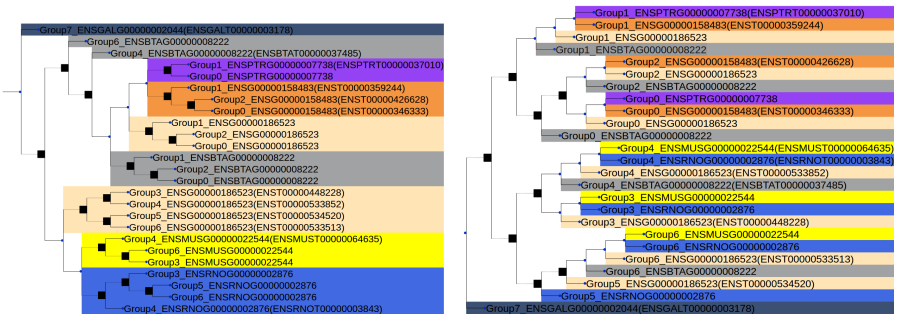


Fig. 6: **Left.** The initial protein tree P' computed using PhyML. Protein lines are colored according to the gene to which they belong (blue: *ENSRNOG00000002876* ; dark blue : *ENSGALG00000002044* ; orange : *ENSG00000158483* ; gray : *ENSBTAG00000008222* ; beige : *ENSG00000186523* ; yellow : *ENSMUSG00000022544*); purple : *ENSPTRG00000007738*. **Right.** The protein supertree P reconstructed using SuperProteinTree. The creation events inferred in the protein trees after reconciliation with the gene tree are represented as square nodes in black colors.

Discussion: As expected, for both gene families, FAM86 and TP63, in the protein tree P' computed using a sequence-based reconstruction method, the proteins of each gene are grouped into complete subtrees. As a consequence, all the creation

Table 4: SuperProteinTree results obtained on $n = 72$ proteins for the gene family TP63. The notation is the same as the one used for Table 3, except that the dataset for TP63 contains 24 initial orthology groups instead of 8 for FAM86.

		γ											
		n/2			**n**			**2n**			**3n**		
α	β	**A**	**B**	**C**	**A**	**B**	**C**	**A**	**B**	**C**	**A**	**B**	**C**
1.0	**0.0**	1	61	14/23	1	62	14/24	1	65	14/24	0	28	14/24
0.8	**0.2**	0	61	14/24	**1**	**28**	**22/24**	0	25	21/24	0	20	19/24
0.6	**0.4**	0	58	14/24	0	51	14/24	0	35	18/24	0	35	18/24
0.4	**0.6**	0	59	14/24	0	45	15/24	0	37	17/24	0	37	17/24
0.2	**0.8**	0	57	14/24	0	52	14/24	0	52	14/24	0	59	14/24
0.0	**1.0**	0	59	14/24	0	58	14/24	0	58	14/24	0	56	14/24

events are located in these subtrees at the bottom of the tree P' and each gene has its specific set of protein creation events not shared by any other gene. However in the protein supertree P reconstructed using our method, the creation events are distributed from the root to the leaves of the tree. This supports the hypothesis of groups of orthologous protein isoforms shared by several extant genes, and originated from ancestral protein creation events.

The results given in Table 3 and 4 show that by giving more weight to structural similarity than to sequence similarity between proteins, SuperProteinTree achieved a better performance in both experiments. Specially, when $\alpha = 0.8$ and $\beta = 0.2$, our method is able to reconstruct a single protein tree displaying the highest ratio of recovered initial orthology groups, with the smallest number of creation events. It is worth noting that in the special case when $\alpha = 1.0$ and $\beta = 0.0$, the SuperProteinTree algorithm is able to reconstruct one protein tree, but with more creation events and less recovered initial orthology groups. This might be due to the fact that the SuperProteinTree algorithm tries to group proteins which have the same structure but are too different in terms of sequence. A similar phenomenon occurs when $\alpha = 0.0$ and $\beta = 1.0$. Again, the number of creations is high and the number of recovered initial orthology groups is low, suggesting that the structure of proteins should not be ignored. These results illustrate the advantage of incorporating both pieces of information into the calculation of distance between proteins.

8. Conclusion

In this work, we have argued the importance of distinguishing gene trees from protein trees, and introduced the notion of protein trees into the framework of reconciliation. We have shown that, just as gene trees are thought of as evolving "inside" a species tree, protein trees evolve "inside" a gene tree, leading to two layers of reconciliation. We provided evidence that, even if each gene in a given

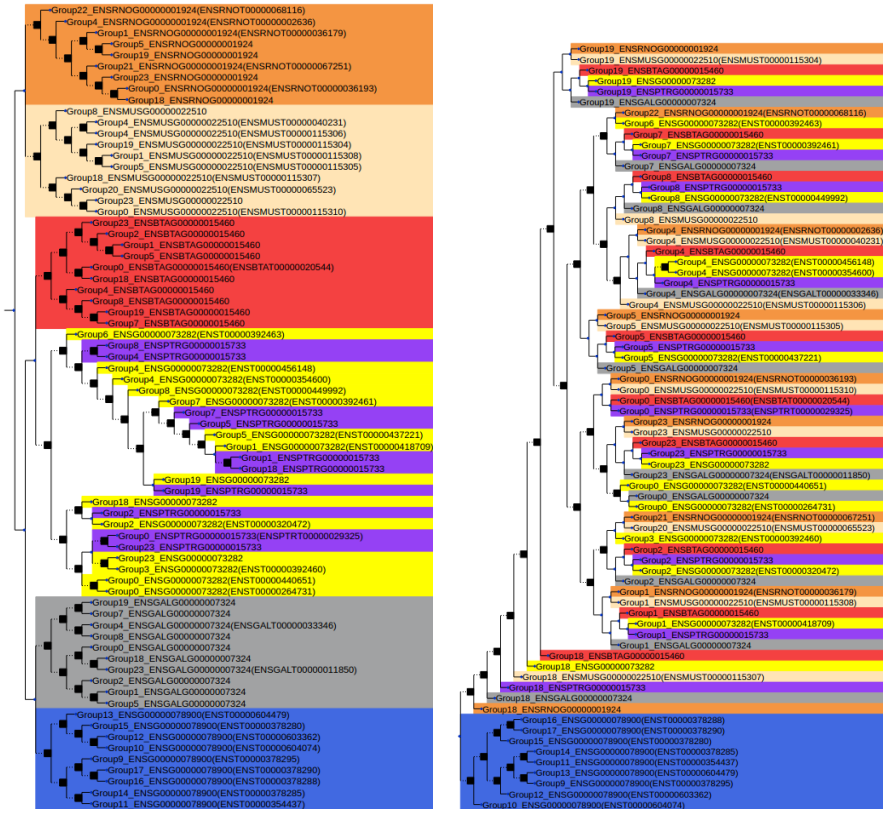


Fig. 7: **Left.** The initial protein tree P' computed using PhyML. Protein lines are colored according to the gene to which they belong.**Right.** The protein supertree P reconstructed using SuperProteinTree. The creation events inferred in the protein trees after reconciliation with the gene tree are represented as square nodes in black colors.

family encodes a single protein, the gene phylogeny does not have to be the same as the protein phylogeny, and may rather behave like a "median" between the protein tree and the species tree in terms of mutation cost. We have also introduced the idea of reconstructing phylogenies from a set of orthology constraints represented as proteins orthology soft-clusters. We have designed an algorithm to reconstruct a protein phylogeny from a set of orthology soft-clusters.

On the algorithmic side, many questions related to the double-reconciliation cost deserve further investigation. For instance, what is the complexity of finding an optimal gene tree in the case that $\mathcal{P} \Leftrightarrow \mathcal{G} \Leftrightarrow \mathcal{S}$? Also, given that the general MinDRGT problem is NP-hard, can the optimal gene tree G be approximated

within some constant factor? Or is the problem fixed-parameter tractable with respect to some interesting parameter, e.g. the number of apparent creations in the protein tree, or the maximum number of proteins per gene? As for the MinDRPGT problem, it remains to explore how the partially labeled protein trees can be used to infer the gene tree. Moreover, we have studied an ideal case where all maximum creation-free protein subtrees could be inferred perfectly. Future work should consider relaxing this assumption by allowing the input subtrees to have missing or superfluous leaves, or to contain errors. Finally, we have designed an algorithm for building a complete phylogeny satisfying a set of orthology soft-clusters in the special case where all orthology soft-clusters are consistent. We defer to a future work the study of the consistency problem for orthology soft-clusters, and the problems of reconstructing a phylogeny from a set of inconsistent orthology soft-clusters.

Acknowledgments

Esaie Kuitche acknowledges the support of Faculty of Sciences of the Université de Sherbrooke.
Manuel Lafond acknowledges the support of the Natural Sciences and Engineering Research Council (NSERC).
Aïda Ouangraoua acknowledges the support of the Canada Research Chairs (CRC), the NSERC and the Fonds de Recherche du Québec - Nature et Technologies (FRQNT).

References

1. Åkerborg Ö, Sennblad B, Arvestad L, Lagergren J, Simultaneous bayesian gene tree reconstruction and reconciliation analysis, *Proceedings of the National Academy of Sciences* **106**(14):5714–5719, 2009.
2. Åkerborg Ö, Sennblad B, Arvestad L, Lagergren J, Simultaneous bayesian gene tree reconstruction and reconciliation analysis, *Proceedings of the National Academy of Sciences* **106**(14):5714–5719, 2009.
3. Barbosa-Morais NL, Irimia M, Pan Q, Xiong HY, Gueroussov S, Lee LJ, Slobodeniuc V, Kutter C, Watt S, Çolak R, *et al.*, The evolutionary landscape of alternative splicing in vertebrate species, *Science* **338**(6114):1587–1593, 2012.
4. Chang WC, Eulenstein O, Reconciling gene trees with apparent polytomies, *COCOON*, Springer, pp. 235–244, 2006.
5. Chauve C, El-Mabrouk N, New perspectives on gene family evolution: Losses in reconciliation and a link with supertrees., *RECOMB*, Springer, pp. 46–58, 2009.
6. Chen K, Durand D, Farach-Colton M, Notung: a program for dating gene duplications and optimizing gene family trees, *Journal of Computational Biology* **7**(3-4):429–447, 2000.
7. Christinat Y, Moret BM, Inferring transcript phylogenies, *BMC bioinformatics* **13**(9):1, 2012.
8. Cunningham F, Amode MR, Barrell D, *et al.*, Ensembl 2015, *Nucleic Acids Research* **43**(D1):D662–D669, 2015.

9. Doyon JP, Ranwez V, Daubin V, Berry V, Models, algorithms and programs for phylogeny reconciliation, *Briefings in bioinformatics* **12**(5):392–400, 2011.
10. Eulenstein O, Huzurbazar S, Liberles DA, Reconciling phylogenetic trees, *Evolution after gene duplication* pp. 185–206, 2010.
11. Goodman M, Czelusniak J, Moore GW, Romero-Herrera A, Matsuda G, Fitting the gene lineage into its species lineage, a parsimony strategy illustrated by cladograms constructed from globin sequences, *Systematic Biology* **28**(2):132–163, 1979.
12. Gorecki P, Eulenstein O, Tiuryn J, Unrooted tree reconciliation: a unified approach, *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)* **10**(2):522–536, 2013.
13. Górecki P, Tiuryn J, Dls-trees: a model of evolutionary scenarios, *Theoretical computer science* **359**(1):378–399, 2006.
14. Guindon S, Gascuel O, A simple, fast, and accurate algorithm to estimate large phylogenies by maximum likelihood, *Systematic biology* **52**(5):696–704, 2003.
15. Hahn MW, Bias in phylogenetic tree reconciliation methods: implications for vertebrate genome evolution, *Genome biology* **8**(7):1, 2007.
16. Hubbard T, Barker D, Birney E, Cameron G, Chen Y, Clark L, Cox T, Cuff J, Curwen V, Down T, *et al.*, The ensembl genome database project, *Nucleic acids research* **30**(1):38–41, 2002.
17. Irimia M, Rukov JL, Penny D, Roy SW, Functional and evolutionary analysis of alternatively spliced genes is consistent with an early eukaryotic origin of alternative splicing, *BMC Evolutionary Biology* **7**(1):188, 2007.
18. Kapustin Y, Souvorov A, Tatusova T, Lipman D, Splign: algorithms for computing spliced alignments with identification of paralogs, *Biology direct* **3**(1):20, 2008.
19. Keren H, Lev-Maor G, Ast G, Alternative splicing and evolution: diversification, exon definition and function, *Nature Reviews Genetics* **11**(5):345–355, 2010.
20. Lafond M, Chauve C, El-Mabrouk N, Ouangraoua A, Gene tree construction and correction using supertree and reconciliation, *IEEE/ACM Transactions on Computational Biology and Bioinformatics (Proceedings of Asia Pacific Bioinformatics Conference APBC'17)* , 2017, dOI 10.1109/TCBB.2017.2720581.
21. Lafond M, Swenson K, El-Mabrouk N, An optimal reconciliation algorithm for gene trees with polytomies, *Algorithms in Bioinformatics* pp. 106–122, 2012.
22. Ma B, Li M, Zhang L, From gene trees to species trees, *SIAM Journal on Computing* **30**(3):729–752, 2000.
23. Nilsen TW, Graveley BR, Expansion of the eukaryotic proteome by alternative splicing, *Nature* **463**(7280):457–463, 2010.
24. Noutahi E, Semeria M, Lafond M, Seguin J, Boussau B, Guguen L, El-Mabrouk N, Tannier E, Efficient gene tree correction guided by genome evolution, *Plos One* , 2016, to appear.
25. Osório J, Evolutionary genetics: Alternative splicing shapes vertebrate evolution, *Nature Reviews Genetics* **16**(10):565–565, 2015.
26. Ranwez V, Harispe S, Delsuc F, Douzery EJ, MACSE: Multiple Alignment of Coding SEquences accounting for frameshifts and stop codons, *PLoS One* **6**(9):e22594, 2011.
27. Rasmussen MD, Kellis M, A bayesian approach for fast and accurate gene tree reconstruction, *Molecular Biology and Evolution* **28**(1):273–290, 2011.
28. Shimodaira H, An approximately unbiased test of phylogenetic tree selection, *Systematic biology* **51**(3):492–508, 2002.
29. Shimodaira H, Hasegawa M, CONSEL: for assessing the confidence of phylogenetoc tree selection, *Bioinformatics* **17**:1246- 1247, 2001.
30. Vilella AJ, Severin J, Ureta-Vidal A, Heng L, Durbin R, Birney E, Ensemblcompara

genetrees: Complete, duplication-aware phylogenetic trees in vertebrates, *Genome research* **19**(2):327–335, 2009.
31. Wu YC, Rasmussen MD, Bansal MS, Kellis M, Treefix: statistically informed gene tree error correction using species trees, *Systematic biology* **62**(1):110–120, 2013.
32. Zambelli F, Pavesi G, Gissi C, Horner DS, Pesole G, Assessment of orthologous splicing isoforms in human and mouse orthologous genes, *BMC genomics* **11**(1):1, 2010.

Esaie Kuitche received the Master's degree in Computer Engineering from École Polytechnique de Yaoundé, Cameroun in 2016. He is a PhD candidate at the Department of Computer Science of Université de Sherbrooke, Québec, Canada, where he develops models and algorithms for the reconstruction of orthology clusters and phylogenies for protein and gene families. He has a scholarship from the Faculty of Science of Université de Sherbrooke.

Manuel Lafond received the PhD degree from the University of Montreal, Canada in 2016. He is now a NSERC postdoctoral fellow in the Department of Mathematics and Statistics at the University of Ottawa, Canada, where he is developing new models and methods to distinguish orthologous genes from paralogous genes. His research interests mainly reside in the intersection of computational biology, algorithms and graph theory.

Aïda Ouangraoua received the PhD degree in Computer Science from University of Bordeaux, France, in 2007. She did her postdoctoral training at Simon Fraser University and Université du Québec à Montréal, Canada. She was a researcher at INRIA Lille, France from 2009 to 2014. She is currently a professor at the Department of Computer Science of Université de Sherbrooke, Québec, Canada, where she holds the Canada Research Chair in Computational and Biological Complexity. Her research interests include Algorithms and Computational Genomics.

Chapitre 4

Simulation de l'évolution des séquences biologiques en considérant l'épissage alternatif

4.1 Introduction

Étant donné l'importance du mécanisme d'épissage alternatif chez les eucaryotes, de nombreux travaux ont été réalisés pour mieux en tenir compte dans les analyses de séquences biologiques. Parmi ces études, on peut distinguer trois grands groupes. Le premier porte sur **l'alignement des séquences épissées :** le but de ces travaux est de proposer de nouvelles approches pour aligner les transcrits et les gènes ayant subi le mécanisme de l'épissage alternatif. Ces méthodes alignent les séquences en prenant en compte leurs structures exon-intron. Au lieu de chercher à optimiser un score d'alignement global entre deux séquences, ces méthodes cherchent à optimiser le score d'alignement dans les régions conservées après épissage (exons). À titre d'exemple, Splign[33] et SpliceFamAlign[31] sont des algorithmes conçus pour aligner un gène avec un transcrit dans le but de détecter et aligner les exons othologues entre les deux séquences. Le second groupe porte sur **l'inférence des phylogénies de gènes et de transcrits :** l'objectif de ces travaux est de proposer de nouvelles méthodes de reconstruction des phylogénies des gènes et de transcrits qui prennent

en compte la structure exon-intron des séquences. Dans les travaux de Christinat et al.[10, 11] et Ait-hamlat et al.[1], on propose des modèles d'évolution des transcrits à l'échelle de leurs exons et on s'en sert par la suite pour reconstruire l'évolution des transcrits. Dans notre article[35], nous proposons un modèle d'évolution des transcrits qui tient compte de la production multiple de transcrits par gène. Le troisième groupe porte sur **l'identification des transcrits orthologues et de la prédiction de la structure des gènes :** le but de ces travaux est d'une part d'utiliser la structure exon-intron des transcrits pour identifier les relations d'orthologies entre les transcrits[4] et d'autre part, pour la prédiction des gènes[6]. Dans ces travaux, les soulève la question d'évaluation expérimentale de ces outils. Pour ce faire, on peut utiliser des données réelles sur l'évolution des séquences épissées, ce qui est difficile à obtenir en pratique. Simuler l'évolution des séquences en incluant le mécanisme de l'épissage alternatif s'avère une bonne alternative ; c'est la contribution de ce chapitre.

Ce chapitre est structuré comme suit : la section 4.2 présente la liste des outils de simulations existants. La section suivante présente les limites des outils existants. Enfin, on présente SimSpliceEvol, l'outil de simulation que nous avons proposé dans le but simuler l'évolution des transcrits alternatifs.

4.2 Outils de simulation de séquences biologiques existants

Dans cette section, sont présentés les différents outils de simulation de l'évolution des séquences biologiques. Le premier modèle présenté est un modèle très simpliste. Les modèles suivants viendront l'enrichir.

4.2.1 Modèle de base

Le modèle de base de simulation de l'évolution de séquences biologiques a pour but de générer un jeu de données simulées très simpliste. Ce modèle a pour principale entrée un arbre binaire enraciné. La première étape est la simulation d'une séquence biologique à la racine. Par la suite, on fait évoluer cette séquence du noeud racine vers les feuilles de l'arbre. Entre deux noeuds, la séquence évolue au travers des opérations

d'insertions, de délétions ou de substitution de nucléotides. On termine ainsi avec un ensemble de séquences aux feuilles résultant de la séquence initiale simulée à la racine qui a évolué. Pour les séquences résultantes aux feuilles de l'arbre, tous les événements évolutifs que le gène a subis sont connus. Cet ensemble d'information peut donc être utilisé comme donnée réelle pour la comparaison avec des données estimées. Les données simulées par ce modèle peuvent être totalement décorrélées des données réelles. Dans la section suivante, nous présentons l'enrichissement de ce modèle dans le but de le rendre plus réaliste.

4.2.2 Méthodes existantes

Cette section regroupe en trois grands points les améliorations des modèles de base de simulation de séquences biologiques.

Données simulées

Afin de refléter la structure réelle des gènes, certains outils distinguent la simulation des séquences codantes[26, 64, 21, 60] et des séquences non codantes[63, 7, 26, 64, 21, 60]. Pour chacun de ces deux types de régions, un modèle spécifique est utilisé. Ceci permet de distinguer d'un côté la simulation des régions codantes qui se fait sur la base des codons et de l'autre côté la simulation des régions non codantes qui se fait sur la base d'un nucléotide. Ainsi, la simulation de ces deux types de séquences est utilisée pour illustrer l'évolution des protéines avec certains outils[63, 26, 64, 21, 60]. Certains outils plus réalistes simulent la structure exon-intron[64, 21, 60] des séquences et même des sites d'épissages[60]. Tous ces outils de simulations génèrent un ensemble de séquences avec leur alignement réel. Le tableau 4.1 illustre une comparaison du type de données simulées.

Traitement des indels

Contrairement au modèle de base dans lequel le nombre d'indels et leur position ne suivent pas de modèle spécifique, des modèles plus réalistes sont proposés. Parmi ces modèles, on peut citer ceux qui simulent un nombre d'indels[63, 7, 49, 26] en fonction de la longueur des séquences. D'autre modèles définissent des probabilités

	ROSE	Dawg	SIMPROT	EvolveAGene3	iSGv2.0	INDELible	PhyloSim
Données simulées							
Structure exon-intron					X	X	X
Sites d'épissages							X
Séquence codante				X	X	X	X
Séquence non codante	X	X		X	X	X	X
Séquence de protéine	X		X	X[a]	X	X	X
Alignement des séquences	X	X	X	X	X	X	X

Tableau 4.1 – Comparaison de 7 outils de simulation sur la base des données simulées. [a] Les méthodes peuvent générer des séquences d'acides aminés, mais la simulation n'est effectuée qu'au niveau des nucléotides

pour choisir le type d'indel (insertion ou délétion)[63, 7, 26, 64, 21, 60]. Certains de ces outils proposent des modèles pour simuler la longueur de chaque indels. Le tableau 4.2 présente un comparatif montrant les outils de simulation courants traitant les indels.

	ROSE	Dawg	SIMPROT	EvolveAGene3	iSGv2.0	INDELible	PhyloSim
Traitement d'Indel							
Continue	X	X	X	X			
Ajustement dynamique de la longueur		X			X	X	X
Suivi des événements					X	X	X
Probabilité de ins et del indépendant	X	X		X	X	X	X
Distribution de longueur empirique		X	X	X	X	X	X

Tableau 4.2 – Comparaison de sept outils de simulation sur la base du traitement des indels

Simulation de l'évolution des séquences

Partant des données simulées, des modèles sont proposés pour simuler l'évolution des composants des séquences. Quelques-uns de ces modèles simulent l'évolution des régions spécifiques des séquences[63, 64, 21, 60]. Un seul de ces modèles [60] simule

l'évolution des sites d'épissage. Le tableau 4.3 présente un comparatif montrant comment les outils de simulation courants traitent l'évolution des séquences biologiques.

	ROSE	Dawg	SIMPROT	EvolveAGene3	iSGv2.0	INDELible	PhyloSim
Évolution simulée							
Sites d'épissage							X
Séquence codante				X	X	X	X
Séquence non codante	X	X		X	X	X	X
Séquence protéique	X		X		X	X	X
Évolution hétérogène (partition)		X	X		X	X	X
Conservation des motifs spécifiques à la lignée					X		
Conservation des motifs spécifiques à la longueur	X				X		
Conservation des motifs spécifiques aux sites					X	X	X

Tableau 4.3 – Comparaison de sept outils de simulation sur la base du traitement de l'évolution des séquences

4.3 Limites des méthodes de simulation de séquences biologiques existantes

Bien que tous les modèles de simulation susmentionnés modélisent les séquences biologiques et leur évolution tout en intégrant plusieurs composants, il existe un certain nombre de limites que nous listerons ci-dessous.

— **Pas de simulation de l'épissage alternatif :** les modèles de simulations que l'on retrouve dans la littérature ne prennent pas en compte le mécanisme de l'épissage alternatif. Ces modèles se basent sur le dogme central de la biologie moléculaire dans lequel on affirmait qu'un gène produit un transcrit. Ainsi, les modèles courants font évoluer une séquence de gène de la racine aux feuilles de l'arbre. Cela représente une limitation actuellement, car l'analyse des génomes eucaryotes prennent désormais compte de ce mécanisme. Puisque ces modèles ne simulent qu'une seule séquence, ils ne peuvent pas être utilisés dans un contexte où l'on s'intéresse aux transcrits épissés.

— **Pas de distinction entre les différents niveaux d'évolution :** les modèles de simulations courants font évoluer une séquence ancestrale de la racine aux feuilles. Cette séquence correspond le plus souvent à la séquence d'un gène. Ce faisant, ces modèles ne définissent pas les différents niveaux d'évolution. Ils se contentent de faire évoluer une séquence. Il est nécessaire de distinguer au moins trois niveaux d'évolution, à savoir : l'évolution des exons-introns, l'évolution des transcrits et l'évolution des gènes. À chaque niveau d'évolution, un modèle spécifique doit être défini. Christinat *et al.*[10, 11] ont proposé un modèle d'évolution des exons. D'après le modèle qu'ils proposent, les exons évoluent soit par gain, par perte ou par duplication. En ce qui concerne l'évolution des transcrits, nous avons proposé un modèle[35]. Ici, les transcrits évoluent à travers des événements de création de transcrits et perte de transcrits. La création des transcrits fait ici allusion au fait qu'un gène a eu la capacité, à un moment donné, de produire un nouveau transcrit. Ceci peut être réalisé par l'un des cinq mécanismes de l'épissage alternatif 1.1.6. Enfin, l'évolution des gènes se fait au travers des événements de pertes et de duplications de gènes.

Afin de pallier les limites des outils existants, nous avons proposé un nouveau modèle de simulation SimSpliceEvol que nous présentons dans la section suivante.

4.4 Article : "SimSpliceEvol : Alternative splicing-aware simulation of biological sequence evolution"

Afin de fournir un outil de simulation d'évolution de séquences biologiques qui prend en compte l'épissage alternatif des transcrits de gènes, nous avons développé SimSpliceEvol. Les deux principaux objectifs de SimSpliceEvol sont d'une part de simuler une évolution réaliste de la structure d'épissage des gènes, et d'autre part, de simuler une évolution réaliste des séquences, en distinguant les séquences d'exons des séquences d'introns. Le processus de simulation de l'évolution de séquences biologiques avec SimSpliceEvol peut se résumer en quatre points. 1) Générer une structure de gène et sa séquence à la racine de l'arbre. 2) Simuler la production d'un ensemble de

transcrit à la racine de l'arbre via le mécanisme de l'épissage alternatif. 3) Simuler l'évolution des gènes et des transcrits d'un noeud parent vers ses noeuds enfants, de la racine vers les feuilles de l'arbre. 4) Conserver l'histoire d'évolution des séquences.

J'ai conçu l'étude avec Pr. Ouangraoua. J'ai développé les algorithmes sous la supervision de Pr. Ouangraoua. J'ai développé le programme et sa documentation, collecté les données, mené les expériences et présenté les résultats lors de la conférence RECOMB-CG'19. J'ai rédigé le manuscrit. Jammali a réalisé les statistiques pour les résultats expérimentaux du manuscrit. Pr. Ouangraoua a révisé de manière critique le manuscrit. Tous les auteurs ont lu et approuvé le manuscrit final. L'article a été publié dans le journal BMC Bioinformatics.

RESEARCH

SimSpliceEvol: Alternative splicing-aware simulation of biological sequence evolution

Esaie Kuitche[1*], Safa Jammali[1,2] and Aïda Ouangraoua[1]

*Correspondence:
Esaie.Kuitche.Kamela@USherbrooke.ca
[1]Department of Computer Science, University of Sherbrooke, 2500 Boulevard de l'Université, J1K2R1 Quebec, CANADA
Full list of author information is available at the end of the article

Abstract

Background: It is now well established that eukaryotic coding genes have the ability to produce more than one type of transcript thanks to the mechanisms of alternative splicing and alternative transcription. Because of the lack of gold standard real data on alternative splicing, simulated data constitute a good option for evaluating the accuracy and the efficiency of methods developed for splice-aware sequence analysis. However, existing sequence evolution simulation methods do not model alternative splicing, and so they can not be used to test spliced sequence analysis methods.

Results: We propose a new method called SimSpliceEvol for simulating the evolution of sets of alternative transcripts along the branches of an input gene tree. In addition to traditional sequence evolution events, the simulation also includes gene exon-intron structure evolution events and alternative splicing events that modify the sets of transcripts produced from genes. SimSpliceEvol was implemented in Python. The source code is freely available at https://github.com/UdeS-CoBIUS/SimSpliceEvol

Conclusion: Data generated using SimSpliceEvol are useful for testing spliced RNA sequence analysis methods such as methods for spliced alignment of cDNA and genomic sequences, multiple cDNA alignment, orthologous exons identification, splicing orthology inference, transcript phylogeny inference, which requires to know the real evolutionary relationships between the sequences.

Keywords: simulation; exon-intron structure; alternative splicing; evolution

Background

Alternative splicing is used by eukaryotic coding genes to diversify their transcript production [1]. Splicing [2] is a mechanism by which a primary transcript from a gene undergoes cutout and ligation steps that lead to the elimination of some segments from the transcript to result in a final mature transcript. The segments conserved in the mature transcript are called exons and those that are eliminated by splicing are called introns. Thus, the exon-intron structure of a eukaryotic gene refers to the succession of alternating exon and intron segments that compose the gene sequence. Alternative splicing allows genes to produce several isoforms of transcripts composed of different combinations of exons [2].

The gain and loss of introns and exons in gene structures along the evolution have been studied in various lineages of eukaryots. Intron loss and gain by unknown mechanisms were detected in several lineages [3, 4, 5]. Exon loss and gain were also

observed in various lineages by mechanisms including genomic deletion, insertion or duplication, and mutational disabling or acquisition of splice sites [6, 7, 8]. It was estimated that approximately 95% of multiexonic human genes give rise to alternative splicing [9]. Evolutionary comparisons of alternative exons and transcripts have shown a significant enrichment for evolutionary conservation, and ancient origins of alternative exons [10, 11]. The functional consequences of changes in exon-intron structure and splicing patterns have been widely documented. It was shown that the majority of alternative splicing events display tissue-dependent variation, and give rise to protein functional changes [12, 13, 14].

Several methods have been developed for the analysis of spliced transcript sequences. These include methods for the computation of spliced alignment between spliced transcript and unspliced genomic sequences [15, 16, 17], the computation of multiple alignment of spliced cDNA sequences [18, 19, 20], the identification of orthologous exons in a set of transcripts, the inference of splicing orthology relations between transcripts [21, 22], the reconstruction of alternative transcript phylogenies [23, 24, 25], the clustering of proteins, transcripts and genes sequences [26, 27, 28, 29], to mention only those. However, the lack of real gold standard data for which the true evolutionary relationships between data are known is an obstacle to the evaluation of the performance of methods for spliced transcript sequences analysis. Thus, sequence evolution simulation constitutes a promising avenue for the generation of simulated benchmark data to test these methods.

A multitude of tools have been developed for the simulation of the evolution of biological sequences [30, 31, 32, 33, 34, 35, 36, 37, 38]. Most of these methods take as main input a guide tree, generate an ancestral sequence at the root of the guide tree, and make this sequence evolve iteratively along the branches of the tree. The simulated evolution events include sequence insertion and deletion (indel) events and substitution events. At the end of the simulated evolution, each leaf of the input guide tree is associated to one sequence for which the full evolutionary history is known. Some sequence evolution simulation tools like PhyloSim [30], indel-Seq-Gen [32] and INDELible [38] simulate the exon-intron structure of genes by defining partitions that evolve under different models and parameters. However the initial exon-intron structure generated at the root the tree can not be modified along the evolution. Some other tools are dedicated to the simulation of raw amino acid sequences [31, 36] or nucleotide sequences [35] evolution without the underlying exon-intron structure of genes. All simulation tools return as result the true alignment of the simulated sequences, which is useful as benchmark data to test sequence analysis methods. However, no existing tool simulates both changes in the exon-intron structure of genes and the alternative splicing mechanisms that drive the evolution of sets of transcripts produced from genes.

In this paper, we present a new simulation tool, called SimSpliceEvol for gene and alternative transcript sequence evolution. SimSpliceEvol simulates events acting on the evolution of the exon-intron structure of genes and alternative splicing events acting on the sets of transcripts produced from genes, in addition to traditional sequence substitution and indel events. SimSpliceEvol takes as input a guide gene tree with branch lengths representing the number of substitutions per site on branches, and generates a set of gene sequences representing a gene family with

the exon-intron structures and the sets of cDNA sequences associated to alternative transcripts of the genes. For all simulated gene and cDNA sequences, the true multiple sequence alignment and the orthology relationships between exons and between transcripts are also given as output. Data produced by SimSpliceEvol can be used to evaluate models and methods for spliced sequence analysis. For instance, in [39], we used it to generate simulated data for the comparison of spliced alignment methods. To the best of our knowledge, SimSpliceEvol is the first sequence evolution simulation tool that integrates the simulation of both the evolution of gene exon-intron structure and alternative splicing events.

The paper is organized as follows. The next section is dedicated to the description of the simulation model of SimSpliceEvol. In the Results section, a comparison of SimSpliceEvol with existing sequence evolution simulation tools is provided. The usefulness of SimSpliceEvol is illustrated by the use of simulated data to compare the performance of methods for multiple cDNA/protein sequence alignment and methods for cDNA/protein clustering.

Materials and Methods

SimSpliceEvol takes as input a guide gene tree with branch lengths, and generates the genes and cDNA sequences at the leaves of the tree with the true evolutionary history of the gene family. The method starts by simulating an ancestral gene sequence with exon-intron structure and a set of alternative cDNA sequences at the root of the tree. Next, two models of evolution are applied jointly to simulate the evolution from the ancestral gene along branches of the guide tree. The first model makes evolve the exon-intron structure of the genes and the resulting sets of alternative cDNA sequences, and the second model is a codon / nucleotide sequence evolution model by substitution and indel events for the exon and intron sequences.

In this section, we first describe how the ancestral gene sequence at the root of the tree is simulated. Next, we describe the simulation models used on the one hand for the evolution of the exon-intron structure of genes and the sets of alternative cDNA, and on the other hand for the evolution of the exon and intron sequences.

Root ancestral gene simulation

In order to generate realistic data, we collected the exon-intron structure, transcript and sequence information from all coding genes from the Ensembl Compara 96 database [40]. These dataset is composed of coding genes and cDNA sequences of 189 eukaryotic species. *Oryzias_latipes* has the highest fraction of genes 1.30%, with a median value of 0.55% for all species. *Homos_sapiens* has the highest fraction of transcripts 2%, with a median value of 0.53% for all species. From this data, we derived the probability distributions of the number of exons per cDNA transcript, the number of alternative cDNA transcripts per gene, the length of exon segments, the length of intron segments, and the pair of dinucleotides at the extremities of an intron segment called splice sites. Figure 1 presents the probability distributions obtained.

The nucleotide sequences of translated exons and introns from the Ensembl coding gene dataset were used to build Markov chains to simulate exon and intron sequences. For exon segments, since the interest is to simulate cDNA sequences

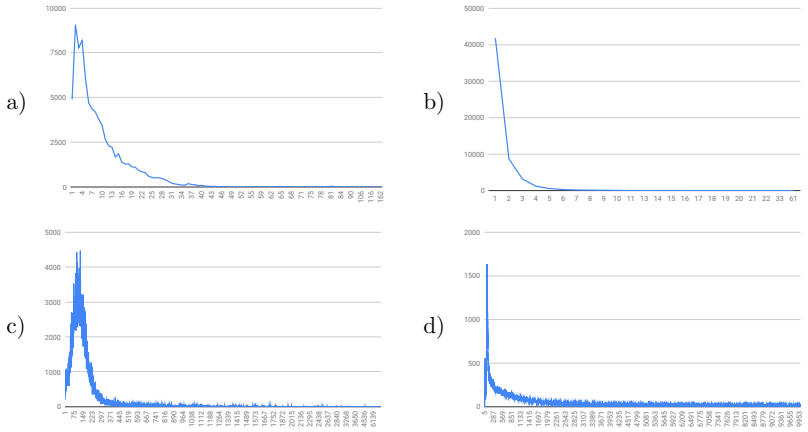


Figure 1 Probability distributions of the composition of gene sequences derived from Ensembl Compara 96 coding genes. (**a**) Number of exons per cDNA transcript (mean: 9.62; std:8.54). (**b**) Number of cDNA transcripts per gene (mean: 1.45; std:1.08). (**c**) Exon length (mean: 170.36; std:258.86). (**d**) Intron length (mean: 3730.30; std:20126.39).

Figure 2 Illustration of a gene structure. A gene structure defined by the succession of alternating exon segments and intron segments with splice sites at their extremities.

composed of translated exons, we derived from the data the probabilities for each nucleotide to be in the first position of a codon, to be in the second position given the nucleotide at the first position, and to be in the third position given the dinucleotide at the two first positions of a codon. The derived probabilities used as transition probabilities for the Markov chains are shown in Table 1.

For intron segments, since the sequences are non-coding, the learning based on codons is not relevant. We simply computed the probabilities for each nucleotide to appear in an intron sequence, and used them to build a zero-order Markov chain. The probabilities of nucleotides are shown in Table 2.

Based on the probability distributions derived for the structure and the sequence composition of genes, the gene exon-intron structure and nucleotide sequence with a set of coding transcripts are generated for the root of the tree. The exon-intron structure of a gene is defined by the succession of alternating exon and intron segments that compose the gene sequence. Intron segments have specific dinucleotides at their extremities called splice sites that are recognized by the splicing machinery. In 98% of the cases, these dinucleotides are the canonical splice sites GT-AG and in 1% of the cases the non-canonical splice sites AC-AT. (See Figure 2 for an illustration) [41, 42].

From the probability distributions (Figure 1), the method first defines the number of exons of the gene. The maximum number m of exons per cDNA is sampled, and the number of exons of the gene is defined as $k_{nbexons} \times m$ where $k_{nbexons} \geqslant 1$ is a user-defined constant. Next, the length of each exon and each intron and the

splice sites associated to each intron are sampled. Once the exon-intron structure is generated, the nucleotide sequence for each exon and each intron is generated using the Markov chains built from the nucleotide probabilities (Tables 1 and 2). An exon sequence is built as a chain of independent codons generated one by one once until the length of the exon is reached. Each codon is built progressively using three Markov chains. The zero-order Markov chain is used to generate the first nucleotide x_1 of the codon, and then the first-order Markov chain is used to generate the second nucleotide x_2, and finally the second-order Markov chains is used to generate the third nucleotide x_3. Note that the model imposes that the length of an exon is a multiple of 3. An intron sequence is built as a chain of independent nucleotides flanked by splice sites chosen from the empirical splice site distribution of 98% for GT-AG, 1% for GC-AG, and 1% for all other types of splice sites. The nucleotides of an intron are generated one by one until the length of the intron is reached.

Alternative splicing allows genes to produce several isoforms of transcripts composed of different combinations of exons [2]. There exist five main types of elementary alternative splicing events that explain the difference between two transcripts produced from a gene. Alternative 3' or 5' splice-site selections occur when two distinct splice sites are used in the intron at the 5' or 3' extremity of an exon. Exon skipping is the alternative inclusion or skipping of an exon in the transcripts. Mutually exclusive exons occur when alternatively one of two successive exons is included but not both. Intron retention is the alternative inclusion or splicing of an intron in the transcripts. Note that two transcripts of a gene may differ by a combination of several alternative splicing events. See Figure 3 for an illustration of the five main types of alternative splicing events and Figure 4 for an illustration of the production of several isoforms of transcripts from a gene by alternative splicing.

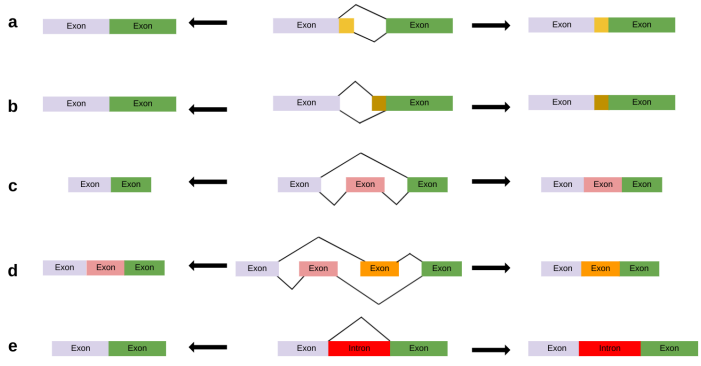


Figure 3 Five main types of alternative splicing events. (**a**,**b**) Alternative 3' and 5' splice-site selections. (**c**) Exon skipping. (**d**) Mutually exclusive exons. (**e**) Intron retention.

The set of alternative transcripts produced at the root of the guide tree is generated first by randomly selecting a subset of the transcripts from the set of all possible isoforms that have a number of exons less or equal to the maximum number m of exons per cDNA. Next, the remaining transcripts are generated by applying alternative splicing events on the transcripts selected randomly.

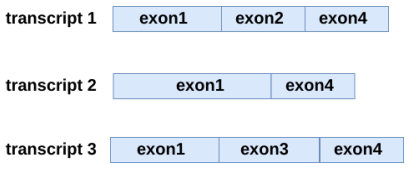


Figure 4 Example of 3 transcripts produced from the gene on Figure 2 by alternative splicing. The difference between Transcripts 1 and 2 is an alternative 3' splice-site selection for Exon 1 and an exon skipping for Exon 2. The difference between Transcripts 1 and 3 is an event of mutually exclusive exons for Exons 2 and 3. The difference between Transcripts 2 and 3 is an alternative 3' splice-site selection for Exon 1 and an exon skipping for Exon 3.

Six user-defined parameters are used to define the proportion of transcripts generated by random selection or by alternative splicing. They are the relative frequencies of transcript generation by random selection tc_{rs}, alternative 5' splice-site selection tc_{a5}, alternative 3' splice-site selection tc_{a3}, exon skipping tc_{es}, mutually exclusive exons tc_{me}, and intron retention tc_{ir}. In the case where $tc_{rs} = 0$, a first transcript is generated by random selection and the other transcripts are generated by alternative splicing. In the case of alternative 3' and 5' splice-site selections, the dinucleotide at the new splice site is modified in order to correspond to a known type of splice site.

Once the root gene and its set of alternative cDNA are simulated, the next step is to simulate their evolution along the branches of the guide tree.

Evolution simulation

SimSpliceEvol combines two models of evolution that are applied conjointly. The first model is the structure evolution model that acts on the evolution the exon-intron structure and the resulting set of transcripts of the gene. The second model is the sequence evolution model that acts on the evolution of the gene sequence. The length of a branch in the input guide tree represents the expected number of substitution events per codon in coding sequences on the branch. To define the expected numbers of any other type of evolutionary events acting on the exon-intron structure, the set of transcripts or the sequence along a branch, we use a linear model, which assumes that all rates are linearly related to the substitution rate.

In the following, we present the two models in two different sections for the sake of clarity, even if there are applied conjointly along branches of the guide tree.

Structure evolution model

The structure evolution model used in SimSpliceEvol is an extension of the Christinat-Moret model of transcript evolution introduced in [24]. The model is divided in two levels of evolution, one acting at the gene level on the gene exon-intron structure and the other at the transcript level on the sets of transcripts produced from a gene. We describe below the evolutionary events included in the model at each level.

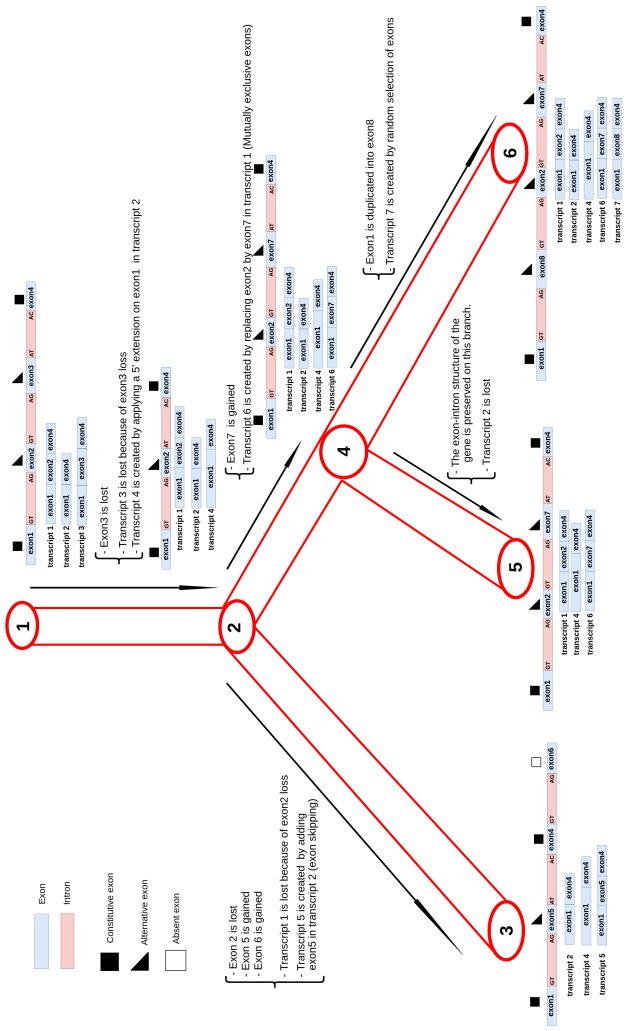


Figure 5 Illustration of SimSpliceEvol model of evolution. Scenario of evolution of the exon-intron structure and the set of transcripts in a gene tree (depicted in red).

Evolutionary events acting on the exon-intron structure of genes. The evolution of the gene exon-intron structure is driven by three elementary events that are the loss, gain and duplication of an exon. The loss of an exon occurs when an exon is removed from the gene exon-intron structure and does not belong anymore to any transcript of the gene. The loss of an exon is implemented as a deletion of the exon segment from the gene sequence. The gain of an exon is the appearance of a new exon segment inside an existing intron. It is implemented by generating a new exon segment and inserting it inside an existing intron segment. The duplication of an exon consists in a tandem duplication of an exon segment and the insertion of an intron segment between the two copies. The application of multiple successive

events along a branch of the guide tree leads to a modification of the exon-intron structure of the gene.

Given a branch of the guide tree with length *codon_subst_rate* representing the expected number of substitution events per codon on the branch, the expected number of exon-intron structure change (EIC) events per exon on the branch is calculated as $k_{eic} \times codon\_subst\_rate$ where k_{eic} is a user-defined constant. Three additional user-defined parameters are used to define the proportion of EIC events acting on the evolution of the exon-intron structure of genes. They are the relative frequencies of exon-intron structure change by exon loss eic_{el}, exon gain eic_{eg}, and exon duplication eic_{ed}, such that the sum of these three relative frequencies equals 1.0. Thus, for example, the overall expected number of exon loss events on a gene having n exons is calculated as $n \times eic_{el} \times k_{eic} \times codon\_subst\_rate$ for a branch of length *codon_subst_rate*.

Evolutionary events acting on the set of transcripts of genes. The evolution of the sets of transcripts produced from genes are driven by the evolution of the gene exon-intron structure, and also by events that take place at the transcript level. Two elementary events can affect the set of transcripts produced from a gene at the transcript level: the loss and the creation of a transcript.

The loss of a transcript occurs a gene stops to produce a given type of transcript due to mutations or a new regulation of the gene expression. The simulation of a transcript loss event consists in stopping the generation of a transcript starting from the node where it is lost. Note that the model makes the assumption that the loss of an exon at the gene level implies a loss of all the transcripts that contain this exon.

The creation of a transcript occurs when a gene starts producing a new type of transcript, i.e., a new combination of exons from its exon-intron structure. Alternative splicing determines which alternative exons are present or absent in each transcript of a gene. The simulation of a transcript creation event in the model consists in generating a new transcript either randomly from the set of all possible isoforms, or by applying an alternative splicing event on an existing transcript (See Figure 3 for an illustration of the five main types of alternative splicing events).

Given a branch of the guide tree with length *codon_subst_rate*, the expected number of transcript change (TC) events per transcript on the branch is calculated as $k_{tc} \times codon\_subst\_rate$ where k_{tc} is a user-defined constant. The six user-defined parameters $tc_{rs}, tc_{a5}, tc_{a3}, tc_{es}, tc_{me}, tc_{ir}$ used for the generation of transcripts at the root of the guide tree are also used as the relative frequencies of TC events by random selection tc_{rs}, alternative 5' tc_{a5} and 3' tc_{a3} splice-site selection, exon skipping tc_{es}, mutually exclusive exons tc_{me}, and intron retention tc_{ir}. An additional user-defined parameter, the relative frequency of TC events by transcript loss tc_{tl} is used to define the relative proportion of TC events by transcript loss, such that the sum of these seven relative frequencies equals 1.0. So, for example, the overall expected number of transcript loss on a gene having n transcripts is calculated as $n \times tc_{tl} \times k_{tc} \times codon\_subst\_rate$ for a branch of length *codon_subst_rate*.

The set of transcripts produced from the exon-intron structure of a gene results in one of the three following states for each exon: absent, alternative, or constitutive.

An exon is absent if it does not belong to any transcript of the gene. An exon is alternative if it may be absent or present in transcripts produced from the gene. A constitutive exon is present in all transcripts produced from the gene. Note that along a branch of the guide tree, an exon can transit from any status to another. SimSpliceEvol does not explicitly integrate the simulation of exon status changes. These changes are induced by the comparison of the sets of transcripts generated at the two extremities of a branch.

Figure 5 illustrates an example of simulation of the evolution of a gene exon-intron structure with the resulting transcripts along branches of a guide tree that has three leaves.

Sequence evolution model

In addition to the evolution of the exon-intron structure and the set of transcripts of genes, the sequence of genes also evolve through insertion and deletion (indel) events and substitution events. A multitude of methods have been developed for the simulation of coding and non-coding sequence evolution. For SimSpliceEvol, we did not develop a new sequence evolution simulation model. We used the same sequence evolution simulation models as indel-Seq-Gen [32] for coding exon and non-coding intron sequences.

Exon sequence evolution model. The model includes codon substitution and indel processes. For each exon segment, codon substitution events along a branch of the guide tree are simulated based on the branch length *codon_subst_rate* that represent the expected number of substitution events per codon on the branch. Each substitution event is generated based on an empirical codon substitution matrix [43] that gives the probability of transition between any two types of codon.

The indels are also simulated based on the branch length *codon_subst_rate*. The expected number of indel events per codon on the branch is calculated as $k_{indel} \times codon\_subst\_rate$ where k_{indel} is a user-defined constant. The length of each indel event is drawn from an empirically derived distribution of indel lengths [44]. The codon sequence of an inserted segment is generated using the Markov chains described in Table 1. Two user-defined parameters are used to define the proportion of insertion and deletion events. The relative frequencies of insertion and deletion events are denoted by ci and cd, such that $ci + cd = 1.0$. Thus, for instance, the overall expected number of codon deletion events on an exon segment composed of n codons is calculated as $n \times cd \times k_{indel} \times codon\_subst\_rate$ for a branch of length *codon_subst_rate*.

Intron sequence evolution model. The model is the same as the exon sequence evolution model, except that the substitution and indels processes are simulated at the nucleotide level, and the expected numbers of substitution and indel events per nucleotide on the guide tree branches are multiplied by a user-defined constant k_{intron}. So, for instance, the overall expected number of nucleotide deletion events on an intron segment of length n is $k_{intron} \times n \times cd \times k_{indel} \times codon\_subst\_rate$ for a branch of length *codon_subst_rate*.

Implementation

SimSpliceEvol generates a gene sequence with exon-intron structure and a set of alternative transcripts at the root of the guide tree according to the probability distributions in Figure 1 and Tables 1 and 2. Next, the program recursively generates the mutations along each branch of the tree from root to leaves, such that each descendant node inherits all the mutations generated along the path between the root and the node. On each branch, exon-intron structure mutations (exon loss, exon gain, and exon duplication) are first performed, then transcript mutations (transcript loss and transcript creation) are performed, and finally sequence mutations (substitution and indels) are performed.

For the simulation of exon-intron structure mutations along a branch (i, j), the method first computes the overall expected number of exon loss events according to the frequency of exon loss and the number of exon at node i, and the exons to be deleted are chosen randomly and removed from the gene. Next, the overall expected number of exon gain events is computed according to the frequency of exon gain and the new number of exons, the insertion positions are randomly chosen, and new exon segments are generated and inserted at these positions. Finally, the overall expected number of exon duplication events is computed according to the frequency of exon duplication and the new number of exons, and the exons to be duplicated are chosen randomly and duplicated.

The simulation of transcript set mutations is done by first performing all transcript loss events and then transcripts creation events by random selection or by one the five alternative splicing event types. For each type of event, the overall expected number of events is adjusted to the number of transcripts at the moment of the simulation.

The simulation of the evolution of each exon and intron sequence is performed independently from the other segments. For exon sequences, the codon evolution model is used, and deletions and insertions are performed before substitutions. As for structure evolution mutations, the overall expected number of a type of event is adjusted to the number of codons in the exon at the moment of the simulation. The position of each event is chosen randomly. For intron sequences, the simulation is performed at the nucleotide level following the same steps as for the exon sequences.

The program outputs the gene sequences at leaves of the guide tree, the set of cDNA sequences for each gene with their exon composition and the location of exons in the gene sequence. SimSpliceEvol also outputs all groups of splicing orthologs that are groups of cDNA transcripts descending from the same ancestral transcript without any alternative splicing events in their evolutionary history from the ancestral transcript. For example in Figure 5, there are six splicing ortholog groups corresponding to Transcripts 1, 2, 4, 5, 6, 7, and the only group with a copy in each gene is the group of Transcript 4. Finally the program keeps track of all the evolutionary events simulated along branches of the tree. This information is used to generate and output the true multiple alignment of all gene and cDNA sequences simulated.

The default values of user-parameters are set as follows, $k_{nbexons} = 1.5$, $k_{eic} = k_{tc} = 5$, $eic_{el} = 0.4$, $eic_{eg} = 0.5$, $eic_{ed} = 0.1$, $tc_{rs} = 0.05$, $tc_{a5} = tc_{a3} = tc_{me} = 0.1$, $tc_{es} = 0.2$, $tc_{ir} = 0.05$, and $tc_{tl} = 0.4$. The default values were chosen to

allow an increase of the numbers of exons and transcripts from the root to the leaves of the guide tree, and also based on results from the literature regarding the levels of alternative splicing among eukaryotes [45, 46, 47]. For instance, it has been shown that intron retention (tc_{ir}) is the rarest type of alternative splicing, whereas exon skipping (tc_{es}) is the more prevalent [45]. The number of user-parameters is intentionally kept large in order to allow the users to simulate and test various frequencies for the evolution events included in the models.

Results

Comparison with existing simulation methods

Table 3 presents a comparison of SimSpliceEvol with existing sequence evolution simulation tools based on criteria used in [32]. The criteria are related to the type of simulated data, simulated evolution and indel treatment. We also consider additional criteria related to exon-intron structure and alternative splicing simulation. Eight simulation methods are compared: ROSE [36], Dawg [35], SIMPROT [31], EvolveAGene3 [37], INDELible [38], indel-Seg-Gen v2.0 (iSGv2.0) [32], PhyloSim [30] and SimSpliceEvol.

The first set of criteria is related to the data generated for the root and the leaves of the guide tree. Four methods, ISGv2, INDELible, Phylosim, and SimSpliceEvol allow the generation of the exon-intron structure of genes, but only the last two allow the generation of splice sites at the extremity of introns. SimSpliceEvol is the only method that generates alternative transcripts and splicing ortholog groups.

The second set of criteria is related to the evolution models integrated in the methods. Among the four methods that allow the generation of gene exon-intron structure, only SimSpliceEvol allows evolving this structure and the resulting set of alternative transcripts. However, it does not include an evolution model for splice sites, as Phylosim does. The main limitation of SimSpliceEvol is that it does not include models for motif conservation, which is important for the simulation of highly diverged gene family evolution. SimSpliceEvol also only allows heterogeneous evolution between exons and introns, but not within exons, or within introns. For the current first version of the method, we chose to focus on the development of models for the evolution of the exon-intron structure and the set of alternative transcripts. Several models of motif conservation and heterogeneous evolution used in existing methods will be integrated in subsequent versions of SimSpliceEvol (See [32] for a review of existing models for simulation with motif conservation and heterogeneous evolution).

The last set of criteria concerns the models for indel treatment. As Dawg, SimSpliceEvol combines a continuous generation of indel events with dynamic length adjustment. The continuous model consists in calculating first the number of events based on the sequence length and then generating the events iteratively. The dynamic length adjustment consists in recalculating the number of events after each change in the length of the sequence in order to avoid under-estimating or over-estimating the number of events. In SimSpliceEvol, deletions and insertions are performed before substitutions. Each series of events is simulated using the continuous model, but the number of events for each series is computed based on the length of the sequence at the moment of the generation.

Application

In order to illustrate the usefulness of SimSpliceEvol for testing spliced sequence analysis methods, we used it to generate 3 datasets of gene families using as guide trees, the following 3 species trees obtained from the Ensembl Compara database [40].

((((bonobo:0.0031, chimpanzee:0.0025):0.0043, human:0.0066):0.0018, gorilla:0.0087):0.0084, orangutan:0.0173);

((rabbit:0.1011, (rat:0.0631, mouse:0.0608):0.0522):0.0019, (gorilla:0.0087, human:0.0084):0.0878);

(chicken:0.1295, (opossum:0.1165, ((mouse:0.1149, human:0.0962):0.0001, cow:0.1136):0.0144):0.0101);

The first tree is a species tree of primates that was used to generate a dataset of 30 gene families with highly similar genes, called the "Small" dataset. The second species tree of primates and rodents was used to generate a dataset called "Medium" containing 30 gene families with moderately similar genes. And finally, the last species tree of amniotes was used to generate a dataset called "Large" of 30 gene families. The average percent sequence identity (PID) of pairs of sequences within the families of the 3 datasets are 72% for Small (ranging between 70 and 79%), 52% for Medium (ranging between 50 and 54%), and 40% for Large (ranging between 37 and 41%).

The 3 simulated datasets were then used to compare the performance of methods for multiple cDNA/protein sequence alignment and cDNA/protein clustering.

Comparing the performance of cDNA alignment methods. For each of the 90 simulated gene families, the set of all cDNA transcript sequences was aligned using the multiple sequence alignment methods MACSE [18], MAFFT [19], and MUSCLE [20]. MACSE is a multiple sequence alignment program that accounts for the underlying codon structure of protein-coding nucleotide sequences. MAFFT is a popular multiple sequence alignment program based on the identification of homologous regions by the fast Fourier transform. MUSCLE is another popular multiple sequence aligner that uses the log-expectation score to speed up its progressive alignment protocol. Note that the set of multiple sequence alignment methods compared here is not exhaustive, as the aim of this experiment is simply to show the usefulness of SimSpliceEvol for the testing of such methods.

Using the real alignments of the simulated datasets as a benchmark, the precision, recall, F-score, and computing time values for each method and each gene family were computed. The results compiled by dataset is presented in Figure 6. The precision measure is the fraction of nucleotide pairs in the estimated alignment that are also in the true alignment. The recall measure is the fraction of nucleotide pairs in the true alignment that are also in the estimated alignment. The F-score is the harmonic mean of precision and recall. We observe that MAFFT is most accurate method with the lowest computing times among the three methods. The accuracy of all methods decreases with the sequence similarity.

Comparing the performance of cDNA clustering methods. The proteins generated from the cDNA sequences of the 90 simulated gene families were clustered using the protein sequence clustering methods CLUSS [48], OrthoFinder [27], and OrthoMCL [28]. CLUSS is an alignment-free method for clustering protein families.

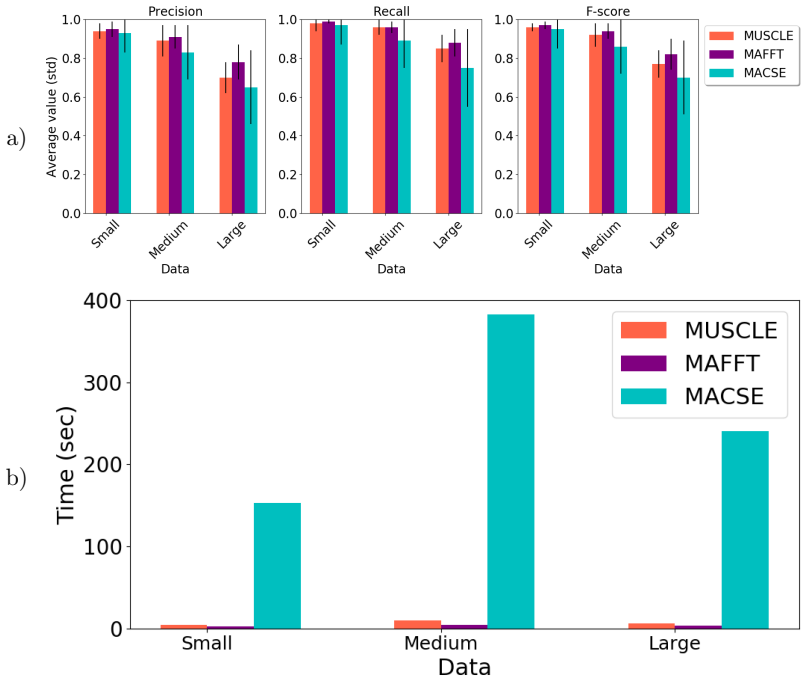


Figure 6 Precision, recall, F-score, and computing time values for multiple sequence alignment methods. Precision, recall, F-score, and computing time values for each dataset (Small, Medium, and Large), and each multiple sequence alignment method (MACSE, MAFFT, and MUSCLE). Precision and recall measures are computed using the true alignment of simulated data as a benchmark.

OrthoFinder is an alignment-based method that infers orthogroups of protein coding genes, by solving the gene length bias in orthogroup inference. OrthoMCL is a popular algorithm for grouping proteins into ortholog groups using pairwise alignment and a Markov Cluster algorithm. It is important to note that none of the three methods was specifically conceived for computing splicing ortholog groups. They were all developed for clustering protein sequences based on their sequences similarities.

Based on the real splicing ortholog groups of the simulated datasets, the Rand index and the computing time for each method and each gene family were calculated. The results are presented in Figure 7. The Rand index is the fraction of pairs of protein sequences that have the same relation in the estimated clustering and the real clustering, either in the same cluster or in different clusters. CLUSS obtains the highest Rand index values with the lowest computing times. OrthoFinder obtains the second-best Rand index values with the highest computing times. OrthoMCL has the lowest Rand index values. The performance of OrthoFinder decreases with the similarity of sequences, while the performances of CLUSS and OrthoMCL are robust to changes in sequence similarity. The average real number of clusters in the tree datasets are 3.71 (std: 0.59) for Small, 9.39 (std: 1.58) for Medium, and 10.90 (std: 1.40) for Large. So, the real number of clusters increases with the dissimilarity

of sequences. We observed that CLUSS always overestimates the number of clusters with a multiplying factor of 1.90 in average for all datasets. OrthoFinder tends to overestimate or underestimate the number of clusters with average multiplying factors of 1.40, 0.26, 0.17 respectively for the Small, Medium and Large datasets. OrthoMCL always underestimates the number of clusters with average multiplying factors of 0.28, 0.13, 0.15 respectively for the Small, Medium and Large datasets.

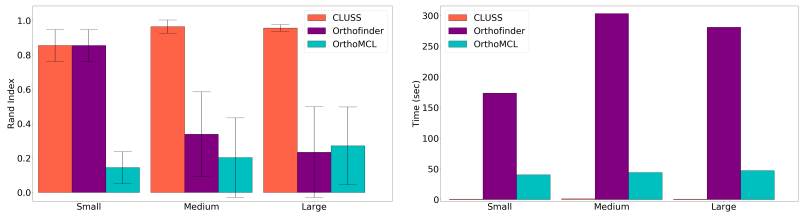


Figure 7 Rand index and computing time values for protein sequence clustering methods. Rand index and computing time values for each dataset (Small, Medium, and Large), and each protein sequence clustering method (CLUSS , OrthoFinder, and OrthoMCL). The Rand index is computed using the true splicing ortholog groups as benchmark.

Conclusion

We present a new sequence evolution simulation method called SimSpliceEvol, that simulates the evolution of the exon-intron structure of genes by exon loss, gain and duplication events, and the evolution of the set of alternative transcripts produced from genes by transcript loss events, and transcript creation events through alternative splicing. SimSpliceEvol also simulates traditional indel and substitution evolution events acting at the sequence level. The main added-value of SimSpliceEvol as compared to all existing sequence evolution simulation methods is the evolution of the exon-intron structure and the set of alternative transcripts. The set of user-parameters allows the users to simulate various frequencies for the evolution events included in the models in order to test various hypotheses regarding exon-intron structure and transcript evolution. Through an application, we show the usefulness of SimSpliceEvol for evaluating the performance of spliced sequence analysis methods like multiple sequence alignment methods, and protein sequence clustering methods.

For the first version of the method, we focus on the development of the models for the evolution of the splicing structure of genes and the resulting transcripts. Several additions will be made in subsequent versions of the method to improve the realism of simulated data. First, the method makes two unrealistic assumptions: (1) independence between the codons of an exon sequence, and (2) the length of exons is always a multiple of 3. The first set of additions will relax these assumptions to generate ancestral exon sequences of any length containing known protein motifs. We will also include models for motif conservation, splice sites evolution, and heterogeneous evolution within exon and intron sequences.

The second set of additions concerns the evolution rates on the branches of the guide tree. Currently, the method assumes a linear relation between the evolution rates of all evolution models included in SimSpliceEvol, i.e., sequence evolution,

exon-intron structure evolution, and set of transcripts evolution. But, to the best of our knowledge, no empirical study of the relationship between the rates of evolution at the sequence and splicing structure levels has been realized yet. To generalize the model, we will extend the method to allow independent evolution rates at different levels, sequence, transcript, and exon-intron structure.

Finally, the current version of SimSpliceEvol does not include an explicit model for the evolution of exons between absent, alternative, and constitutive states. Future versions will include an explicit model for exon status changes, and the set of exon-intron structure evolution events will be extended to include the loss and gain of an intron.

List of abbreviations
DNA: Deoxyribonucleic acid, cDNA: complementary DNA, CDS:Coding DNA sequence.

Declarations
Ethics approval and consent to participate
Not applicable

Consent for publication
Not applicable

Availability of data and materials
Source code available at: https://github.com/UdeS-CoBIUS/SimSpliceEvol Web server: https://simspliceevol.cobius.usherbrooke.ca

Competing interests
The authors declare that they have no competing interests.

Funding
This work was supported by the BEST scholarship program from the Faculty of Science of University of Sherbrooke, the Canada Research Chairs program (CRC Tier2 Grant 950-230577), and the Natural Sciences and Engineering Research Council of Canada (NSERC Discovery Grant RGPIN-2017-05552). Publication costs are funded by the CRC Tier2 Grant 950-230577.

Author's contributions
EK and AO conceived the study and its design. EK wrote the program and its documentation, collected the data, ran the experiments, and presented the results at the conference RECOMB-CG'19. EK drafted the manuscript. SJ realized the figures for the manuscript. AO critically revised the manuscript. All authors read and approved the final manuscript.

Acknowledgments
The authors thank the anonymous reviewers for their helpful comments that contributed to improving the final version.

Author details
[1]Department of Computer Science, University of Sherbrooke, 2500 Boulevard de l'Université, J1K2R1 Quebec, CANADA. [2]Department of Biochemistry, University of Sherbrooke, 3001 12e avenue Nord, J1H5N4 Quebec, CANADA.

References
1. Keren, H., Lev-Maor, G., Ast, G.: Alternative splicing and evolution: diversification, exon definition and function. Nature Reviews Genetics **11**(5), 345 (2010)
2. Graveley, B.R.: Alternative splicing: increasing diversity in the proteomic world. Trends in Genetics **17**(2), 100–107 (2001)
3. Kiontke, K., Gavin, N.P., Raynes, Y., Roehrig, C., Piano, F., Fitch, D.H.: Caenorhabditis phylogeny predicts convergence of hermaphroditism and extensive intron loss. Proceedings of the National Academy of Sciences **101**(24), 9003–9008 (2004)
4. Nielsen, C.B., Friedman, B., Birren, B., Burge, C.B., Galagan, J.E.: Patterns of intron gain and loss in fungi. PLOS biology **2**(12), 422 (2004)
5. Jeffares, D.C., Mourier, T., Penny, D.: The biology of intron gain and loss. Trends in Genetics **22**(1), 16–22 (2006)
6. Alekseyenko, A.V., Kim, N., Lee, C.J.: Global analysis of exon creation versus loss and the role of alternative splicing in 17 vertebrate genomes. RNA **13**(5), 661–670 (2007)

7. Kondrashov, F.A., Koonin, E.V.: Origin of alternative splicing by tandem exon duplication. Human Molecular Genetics **10**(23), 2661–2669 (2001)
8. Merkin, J.J., Chen, P., Alexis, M.S., Hautaniemi, S.K., Burge, C.B.: Origins and impacts of new mammalian exons. Cell Reports **10**(12), 1992–2005 (2015)
9. Pan, Q., Shai, O., Lee, L.J., Frey, B.J., Blencowe, B.J.: Deep surveying of alternative splicing complexity in the human transcriptome by high-throughput sequencing. Nature Genetics **40**(12), 1413 (2008)
10. Yeo, G.W., Van Nostrand, E., Holste, D., Poggio, T., Burge, C.B.: Identification and analysis of alternative splicing events conserved in human and mouse. Proceedings of the National Academy of Sciences **102**(8), 2850–2855 (2005)
11. Xing, Y., Lee, C.: Alternative splicing and rna selection pressure—evolutionary consequences for eukaryotic genomes. Nature Reviews Genetics **7**(7), 499 (2006)
12. Ellis, J.D., Barrios-Rodiles, M., Çolak, R., Irimia, M., Kim, T., Calarco, J.A., Wang, X., Pan, Q., O'Hanlon, D., Kim, P.M., *et al.*: Tissue-specific alternative splicing remodels protein-protein interaction networks. Molecular Cell **46**(6), 884–892 (2012)
13. Kalsotra, A., Cooper, T.A.: Functional consequences of developmentally regulated alternative splicing. Nature Reviews Genetics **12**(10), 715 (2011)
14. Blencowe, B.J.: The relationship between alternative splicing and proteomic complexity. Trends in Biochemical Sciences **42**(6), 407–408 (2017)
15. Bu, J., Chi, X., Jin, Z.: Hsa: a heuristic splice alignment tool. BMC Systems Biology **7**(2), 10 (2013)
16. Huang, S., Zhang, J., Li, R., Zhang, W., He, Z., Lam, T.-W., Peng, Z., Yiu, S.-M.: Soapsplice: genome-wide ab initio detection of splice junctions from rna-seq data. Frontiers in Genetics **2**, 46 (2011)
17. Kapustin, Y., Souvorov, A., Tatusova, T., Lipman, D.: Splign: algorithms for computing spliced alignments with identification of paralogs. Biology Direct **3**(1), 20 (2008)
18. Ranwez, V., Douzery, E.J., Cambon, C., Chantret, N., Delsuc, F.: Macse v2: toolkit for the alignment of coding sequences accounting for frameshifts and stop codons. Molecular Biology and Evolution **35**(10), 2582–2584 (2018)
19. Katoh, K., Standley, D.M.: Mafft multiple sequence alignment software version 7: improvements in performance and usability. Molecular Biology and Evolution **30**(4), 772–780 (2013)
20. Edgar, R.C.: Muscle: multiple sequence alignment with high accuracy and high throughput. Nucleic Acids Research **32**(5), 1792–1797 (2004)
21. Zambelli, F., Pavesi, G., Gissi, C., Horner, D.S., Pesole, G.: Assessment of orthologous splicing isoforms in human and mouse orthologous genes. BMC Genomics **11**(1), 534 (2010)
22. Blanquart, S., Varré, J.-S., Guertin, P., Perrin, A., Bergeron, A., Swenson, K.M.: Assisted transcriptome reconstruction and splicing orthology. BMC Genomics **17**(10), 786 (2016)
23. Kuitche, E., Lafond, M., Ouangraoua, A.: Reconstructing protein and gene phylogenies using reconciliation and soft-clustering. Journal of Bioinformatics and Computational Biology **15**(06), 1740007 (2017)
24. Christinat, Y., Moret, B.M.: Inferring transcript phylogenies. BMC Bioinformatics **13**(9), 1 (2012)
25. Christinat, Y., Moret, B.M.: A transcript perspective on evolution. IEEE/ACM Transactions on Computational Biology and Bioinformatics **10**(6), 1403–1411 (2013)
26. Vilella, A.J., Severin, J., Ureta-Vidal, A., Heng, L., Durbin, R., Birney, E.: Ensemblcompara genetrees: Complete, duplication-aware phylogenetic trees in vertebrates. Genome Research **19**(2), 327–335 (2009)
27. Emms, D.M., Kelly, S.: Orthofinder: solving fundamental biases in whole genome comparisons dramatically improves orthogroup inference accuracy. Genome Biology **16**(1), 157 (2015)
28. Li, L., Stoeckert, C.J., Roos, D.S.: Orthomcl: identification of ortholog groups for eukaryotic genomes. Genome Research **13**(9), 2178–2189 (2003)
29. Kelil, A., Wang, S., Brzezinski, R., Fleury, A.: Cluss: clustering of protein sequences based on a new similarity measure. BMC Bioinformatics **8**(1), 286 (2007)
30. Sipos, B., Massingham, T., Jordan, G.E., Goldman, N.: Phylosim-monte carlo simulation of sequence evolution in the r statistical computing environment. BMC Bioinformatics **12**(1), 104 (2011)
31. Pang, A., Smith, A.D., Nuin, P.A., Tillier, E.R.: Simprot: using an empirically determined indel distribution in simulations of protein evolution. BMC Bioinformatics **6**(1), 236 (2005)
32. Strope, C.L., Abel, K., Scott, S.D., Moriyama, E.N.: Biological sequence simulation for testing complex evolutionary hypotheses: indel-seq-gen version 2.0. Molecular Biology and Evolution **26**(11), 2581–2593 (2009)
33. Tufféry, P.: Cs-pseq-gen: simulating the evolution of protein sequence under constraints. Bioinformatics **18**(7), 1015–1016 (2002)
34. Kosiol, C., Holmes, I., Goldman, N.: An empirical codon model for protein sequence evolution. Molecular Biology and Evolution **24**(7), 1464–1479 (2007)
35. Cartwright, R.A.: Dna assembly with gaps (dawg): simulating sequence evolution. Bioinformatics **21**(Suppl_3), 31–38 (2005)
36. Stoye, J., Evers, D., Meyer, F.: Rose: generating sequence families. Bioinformatics (Oxford, England) **14**(2), 157–163 (1998)
37. Hall, B.G.: Simulating dna coding sequence evolution with evolveagene 3. Molecular Biology and Evolution **25**(4), 688–695 (2008)
38. Fletcher, W., Yang, Z.: Indelible: a flexible simulator of biological sequence evolution. Molecular Biology and Evolution **26**(8), 1879–1888 (2009)
39. Jammali, S., Aguilar, J.-D., Kuitche, E., Ouangraoua, A.: Splicedfamalign: Cds-to-gene spliced alignment and identification of transcript orthology groups. BMC Bioinformatics **20**(3), 133 (2019)
40. Herrero, J., Muffato, M., Beal, K., Fitzgerald, S., Gordon, L., Pignatelli, M., Vilella, A.J., Searle, S.M., Amode, R., Brent, S., et al.: Ensembl comparative genomics resources. Database **2016** (2016)
41. BinEssa, H.A., Zou, M., Al-Enezi, A.F., Alomrani, B., Al-Faham, M.S., Al-Rijjal, R.A., Meyer, B.F., Shi, Y.: Functional analysis of 22 splice-site mutations in the phex, the causative gene in x-linked dominant hypophosphatemic rickets. Bone **125**, 186–193 (2019)

42. Parada, G.E., Munita, R., Cerda, C.A., Gysling, K.: A comprehensive survey of non-canonical splice sites in the human transcriptome. Nucleic Acids Research **42**(16), 10564–10578 (2014)
43. Schneider, A., Cannarozzi, G.M., Gonnet, G.H.: Empirical codon substitution matrix. BMC Bioinformatics **6**(1), 134 (2005)
44. Chang, M.S., Benner, S.A.: Empirical analysis of protein insertions and deletions determining parameters for the correct placement of gaps in protein sequence alignments. Journal of Molecular Biology **341**(2), 617–631 (2004)
45. Kim, E., Magen, A., Ast, G.: Different levels of alternative splicing among eukaryotes. Nucleic Acids Research **35**(1), 125–131 (2006)
46. Barash, Y., Calarco, J.A., Gao, W., Pan, Q., Wang, X., Shai, O., Blencowe, B.J., Frey, B.J.: Deciphering the splicing code. Nature **465**(7294), 53 (2010)
47. Barbosa-Morais, N.L., Irimia, M., Pan, Q., Xiong, H.Y., Gueroussov, S., Lee, L.J., Slobodeniuc, V., Kutter, C., Watt, S., Çolak, R., *et al.*: The evolutionary landscape of alternative splicing in vertebrate species. Science **338**(6114), 1587–1593 (2012)
48. Kelil, A., Wang, S., Brzezinski, R.: Cluss2: an alignment-independent algorithm for clustering protein families with multiple biological functions. International Journal of Computational Biology and Drug Design **1**(2), 122–140 (2008)

a)

x_1 / Start	A	C	T	G
∅	0.24	0.31	0.19	0.25

b)

x_2 / x_1	A	C	T	G
A	0.21	0.28	0.21	0.30
C	0.27	0.38	0.21	0.14
T	0.12	0.35	0.20	0.33
G	0.29	0.30	0.19	0.22

c)

x_3 / x_1x_2	A	C	T	G
AA	0.16	0.35	0.13	0.36
AC	0.22	0.49	0.14	0.15
AT	0.02	0.41	0.14	0.43
AG	0.12	0.55	0.20	0.13
CA	0.09	0.22	0.12	0.57
CC	0.26	0.35	0.28	0.11
CT	0.07	0.26	0.12	0.55
CG	0.17	0.37	0.20	0.26
TA	0.01	0.64	0.34	0.01
TC	0.20	0.48	0.20	0.12
TT	0.10	0.48	0.22	0.20
TG	0.05	0.41	0.30	0.24
GA	0.21	0.25	0.18	0.36
GC	0.15	0.55	0.18	0.12
GT	0.08	0.30	0.16	0.46
GG	0.19	0.39	0.13	0.29

Table 1 Markov chain transition probabilities: probability for a nucleotide A, C, T or G to be: a) in the first position x_1 of a codon, b) in the second position x_2 of a codon given the nucleotide at x_1, and c) in the third position x_3 of a codon given the dinucleotide at x_1x_2.

A	**C**	**T**	**G**
0.29	0.20	0.30	0.21

Table 2 Probability for a nucleotide A, C, T or G to appear in an intron.

	ROSE	Dawg	SIMPROT	EvolveAGene3	iSGv2.0	INDELible	PhyloSim	SimSpliceEvol
Data simulated								
Exon-intron structure					X	X	X	X
Splice sites							X	X
Alternative transcripts								**X**
Coding sequence				X	X	X	X	X
Non-coding sequence	X	X		X	X	X	X	X
Protein sequence	X		X	X[a]	X	X	X	X[a]
Sequence alignment	X	X	X	X	X	X	X	X
Splicing ortholog groups								**X**
Evolution simulated								
Exon-intron structure								**X**
Splice sites							X	
Alternative transcripts								**X**
Coding sequence				X	X	X	X	X
Non-coding sequence	X	X		X	X	X	X	X
Protein sequence	X		X		X	X	X	
Heterogeneous evolution (partition)		X	X		X	X	X	X[b]
lineage-specific motif conservation					X			
Length-specific motif conservation	X				X			
Site-specific motif conservation					X	X	X	
Indel treatment								
Continuous	X	X	X	X				X
Dynamic length adjustment		X			X	X	X	X
Event tracking					X	X	X	X
Probability of ins and del independent	X	X		X	X	X	X	X
Empirical length distribution		X	X	X	X	X	X	X

Table 3 Comparison of 8 sequence simulation methods. [a] The methods can generate amino acid sequences, but the simulation is done only at the nucleotide level. [b] The method defines partitions that evolve under two distinct models for exon and intron, but within an exon or an intron segment, the evolution is homogeneous. Features that are specific to SimSpliceEvol are indicated in bold characters.

Chapitre 5

Algorithme pour la segmentation de transcrits et pour la construction d'arbres de gènes

5.1 Introduction

Dans les chapitres 2 et 3, nous avons effectué une revue de la littérature concernant les principales méthodes de reconstruction d'arbres de gènes. Dans le chapitre 3, nous avons également proposé une méthode pour la correction des arbres de gènes de Ensembl[66]. Le présent chapitre décrit une méthode de reconstruction d'arbres de gènes. La précision des arbres de gènes reconstruits repose sur deux méthodes sous-jacentes qui influencent grandement la qualité des arbres obtenus. D'abord, une nouvelle mesure de similarité entre séquences épissées est définie. Puis, les séquences biologiques sont regroupées suivant cette mesure de similarité selon une représentation vectorielle des séquences et une approche de segmentation floue. Enfin, l'arbre de gènes est reconstruit.

5.2 Mesure de similarités existantes entre séquences biologiques

L'évaluation de la similarité des séquences biologiques est une étape de base dans l'analyse de ces dernières. La similarité permet par exemple d'identifier les séquences homologues qui sont par la suite utilisées dans la reconstruction des phylogénies de transcrits, de protéines, de gènes et d'espèces. Il existe plusieurs définitions de mesure de similarité entre paires de séquences. La majeure partie de ces mesures de similarités est basée sur les séquences. Il en existe également quelques-unes qui exploitent des connaissances sur les structures 2D et 3D des séquences, mais cette information est rarement disponible. Dans cette thèse, nous nous intéressons particulièrement aux mesures des similarités basées sur les séquences.

5.2.1 Mesure de similarité sans alignement

Les mesures de similarités qui ne requièrent pas l'alignement préalable des séquences[67] déterminent la similarité sur la base des motifs partagés entre ces séquences. Elles posent l'hypothèse que deux séquences sont d'autant plus proches si elles partagent des motifs communs. Ces méthodes peuvent se résumer en deux principaux points : premièrement, identifier les motifs de chaque séquence et deuxièmement, évaluer la similarité des séquences en fonction des motifs communs. Il existe plusieurs variantes de ce type de mesures de similarités. Ci-après, nous présentons la mesure de similarité basée sur la fréquence des mots qui illustre bien l'idée générale de cette approche.

Mesure de similarité basée sur la fréquence des mots

Cette mesure de similarité se base sur la fréquence des mots de longueur fixe[3] pour déterminer la similarité entre les séquences.

Définition d'un mot dans une séquence : Étant donnée une séquence X, de longueur n sur un alphabet fini A de cardinalité r, un $L-tuple$ est un segment de longueur L de X. L'ensemble W_L est constitué de tous les $L-tuple$ possibles de A.

$$W_L = \{w_{L1}, w_{L2}, ..., w_{Lk}\}, \tag{1}$$

Une fois l'ensemble W_L obtenu, l'identification des $L-tuples$ appartenant à X est obtenu en comptant le nombre d'occurrences avec chevauchement par ce saut de 1 de $L-tuples$ de X dans W_L. La séquence X est examinée de la position 1 à la position $n-L+1$ afin d'extraire ses $L-tuple$. c_L^X est donc un vecteur qui contient le nombre d'occurrences de chaque $L-tuple$ dans X noté P_{Li}^X.

$$c_L^X = \{P_{L1}^X, P_{L2}^X, ..., P_{LK}^X\}, \tag{2}$$

Déterminer la (di)similarité entre deux vecteurs c_L^X et c_L^Y peut s'effectuer en calculant la distance euclidienne ou toute autre distance entre ces deux vecteurs. S'il s'agit de la distance euclidienne, elle s'exprime comme suit :

$$D(c_L^X, c_L^Y) = \sqrt{\sum_{i=1}^{k}(c_{Li}^X - c_{Li}^Y)^2}.$$

Exemple : soit deux séquences $X = abaababb$ et $Y = baabbaba$ sur l'alphabet $\{a, b\}$. On suppose que $L = 3$. $W_L = \{aaa; aab; aba; abb; baa; bab; bba; bbb\}$. $c_3^X = (0; 1; 2; 1; 1; 1; 0; 0)$ et $c_3^Y = (0; 1; 1; 1; 1; 1; 1; 0)$. La distance euclidienne entre X et Y est $D(X, Y) = \sqrt{2}$.

5.2.2 Mesure de similarité basée sur l'alignement

Tel que présenté au chapitre 1, il existe trois principaux types d'alignement de séquences à savoir : l'alignement global, l'alignement local et l'alignement semi-global. Les mesures de similarités basées sur l'alignement[24] utilisent ces dernières pour déterminer une valeur qui représente la similarité entre les séquences. Afin de définir cette valeur, on se fixe au préalable des poids pour chaque type d'événements repésenté dans un alignement[39]. Par exemple : Dans une colonne de l'alignement de deux séquences, si les résidus sont identiques, on donne un coût de +2, s'ils sont différents, on donne un coût de -1, s'il y a un écart on donne un coût de -2. Étant donné deux séquences $s1 = TTCTTGA$ et $s2 = ATCCTACGA$, le score de similarité entre $s1$ et $s2$ est déduit de l'alignement dont le score est le plus élevé. La figure 5.1 illustre les scores de chaque colonne sur l'alignement de $s1$ et $s2$. Le score d'alignement global encore appelé mesure de similarité est donc $2+2-2-2+2+2-2-1+2+2=5$.

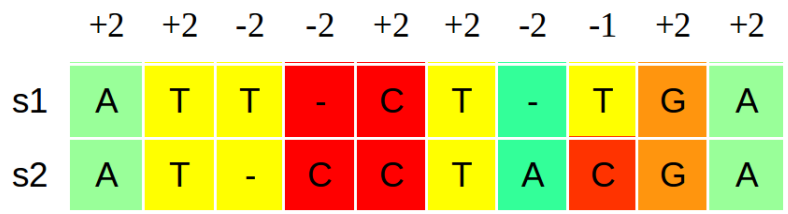


Figure 5.1 – Illustration de l'alignement de séquences et la déduction du coût de similarité. Le coût de similarité entre s1 et s2 est la somme des coûts de chaque colonne. Cette somme dans cet exemple donne 5.

La méthode de calcul que nous avons utilisée dans cet exemple est appelée somme des paires, c'est la méthode la plus utilisée.

5.3 Limites des mesures de similarités existantes

Les mesures de similarités basées ou non sur l'alignement des séquences sont idéales lorsqu'on suppose que la similarité entre deux séquences réside uniquement dans la similarité entre les motifs de ces séquences. Dans les séquences épissées, il existe d'autres caractéristiques structurales qui peuvent être conservées, en plus d'autres motifs. Par exemple, pour un ensemble de transcrits de gènes homologues, la conservation de phases de lecture, des sites d'épissages et des événements d'épissages subits par chaque exon est très importante. Ces caractéristiques ont une influence sur les fonctions des transcrits. Ainsi donc, la prise en compte de ces caractéristiques permettrait d'obtenir une mesure de similarité plus précise.

5.4 Limites des méthodes existantes de segmentation de séquences

La première limite des méthodes de segmentation de transcrits alternatifs résulte de la mesure de similarité qu'elles requièrent. Les mesures de similarité utilisées sont basées uniquement sur la comparaison des séquences de transcrits. Ces méthodes ignorent toutes les informations sur la conservation des structures d'épissage des trans-

crits. La seconde limite des méthodes de segmentation vient du fait que la plupart de ces méthodes dépendent des paramètres fournis par l'utilisateur. Un de ces paramètres, dans le cas de la segmentation non supervisée, est le nombre de groupes idéal pour regrouper les séquences. Dans le cas de la segmentation hiérarchique, l'utilisateur choisit le nombre de groupes final. Dans les méthodes de segmentation par partitionnement comme OrthoMCL, un indice d'inflexion est utilisé pour définir le nombre de groupe. Toutes ces conditions rendent le résultat très dépendant des paramètres fixés par l'utilisateur. L'idéal est d'avoir un modèle qui prend en compte l'information contenue dans les transcrits produits par les gènes, afin de déterminer le nombre optimal de groupes, puis la répartition des transcrits dans ces groupes.

5.5 Article : «Choosing representative proteins based on splicing structure similarity improves the accuracy of gene tree reconstruction »

Afin de pallier aux limites des méthodes existantes de segmentation de séquences, à savoir l'inexploitation des informations sur la conservation des structures d'épissage et la nécessité de paramètres fournis par l'utilisateur, nous proposons une alternative pour la construction des arbres de gènes. Nous présentons une nouvelle mesure de similarité entre transcrits alternatifs combinant une mesure de similarité de la structure d'épissage et une mesure de similarité de la séquence. Ensuite, une méthode de segmentation floue est proposée pour le regroupement de transcrits alternatifs en groupe de forte homologie.

Pr. Ouangraoua a conçu l'étude. J'ai développé les algorithmes sous la supervision de Pr. Ouangraoua et de Pr Wang. Degen a développé le module de collection de données du programme. J'ai développé le programme et sa documentation, collecté les données, mené les expériences. J'ai rédigé le manuscrit. Pr. Ouangraoua et Pr. Wang ont révisé de manière critique le manuscrit. Tous les auteurs ont lu et approuvé le manuscrit final. L'article a été soumis à BioRXiv.

Choosing representative proteins based on splicing structure similarity improves the accuracy of gene tree reconstruction

Esaie Kuitche Kamela[1,*], Marie Degen[1], Shengrui Wang[1], Aïda Ouangraoua[1,*]

[1] Department of Computer Sicence, University of Sherbrooke, Quebec, J1K 2R1, Canada

*Correspondence to be sent to: Department of Computer Sicence, University of Sherbrooke, 2500 Boul. de l'Université Sherbrooke, Quebec, J1K 2R1, Canada.
E-mail: esaie.kuitche.kamela@usherbrooke.ca;aida.ouangraoua@usherbrooke.ca

Abstract. Constructing accurate gene trees is important, as gene trees play a key role in several biological studies, such as species tree reconstruction, gene functional analysis and gene family evolution studies. The accuracy of these studies is dependent on the accuracy of the input gene trees. Although several methods have been developed for improving the construction and the correction of gene trees by making use of the relationship with a species tree in addition to multiple sequence alignment, there is still a large room for improvement on the accuracy of gene trees and the computing time. In particular, accounting for alternative splicing that allows eukaryote genes to produce multiple transcripts/proteins per gene is a way to improve the quality of multiple sequence alignments used by gene tree reconstruction methods. Current methods for gene tree reconstruction usually make use of a set of transcripts composed of one representative transcript per gene, to generate multiple sequence alignments which are then used to estimate gene trees. Thus, the accuracy of the estimated gene tree depends on the choice of the representative transcripts. In this work, we present an alternative-splicing-aware method called Splicing Homology Transcript (SHT) method to estimate gene trees based on wisely selecting an accurate set of homologous transcripts to represent the genes of a gene family. We introduce a new similarity measure between transcripts for quantifying the level of homology between transcripts by combining a splicing structure-based similarity score with a sequence-based similarity score. We present a new method to cluster transcripts into a set of splicing homology groups based on the new similarity measure. The method is applied to reconstruct gene trees of the Ensembl database gene families, and a comparison with current EnsemblCompara gene trees is performed. The results show that the new approach improves gene tree accuracy thanks to the use of the new similarity measure between transcripts. An implementation of the method as well as the data used and generated in this work are available at https://github.com/UdeS-CoBIUS/SplicingHomologGeneTree/.

Keywords: Gene tree, Phylogenetics, Algorithms, Alternative splicing, Splicing homology, Fuzzy C-means

1 Introduction

In the last two decades, several phylogeny methods based on multiple alignment of homologous sequences have been developed and used for gene tree reconstruction (eg. PhyML (Guindon and Gascuel 2003), RAxML (Stamatakis 2006), PhyloBayes (Lartillot and Philippe 2004)). It has been shown that the gene trees reconstructed solely based on multiple sequence alignments often contain many errors and uncertainties, due to numerous reasons which include the quality of gene annotations, the quality of multiple sequence alignments, the quantity of substitutions in the alignments, and the accuracy of the phylogeny reconstruction methods (Hahn 2007; Rasmussen and Kellis 2007). To overcome these limitations, the development of phylogenomics methods which make use a known species tree in addition to multiple sequence alignments has been fruitful to improve the accuracy of reconstructed gene trees (eg. TreeBeST (Schreiber et al. 2014), NOTUNG (Chen, Durand, and Farach-Colton 2000), ALE (Szöllősi et al. 2013), GSR (Åkerborg et al. 2009), TERA (Scornavacca, Jacox, and Szöllősi 2015), SPIMAP (Rasmussen and Kellis 2011), GIGA (Thomas 2010)). Several methods for gene tree correction guided by species tree have also been developed (eg. ProfileNJ (Noutahi et al. 2016), TreeFix (Wu et al. 2013), TreeFix-DTL (Bansal et al. 2015), LabelGTC (El-Mabrouk and Ouangraoua 2017), RefineTree(Lafond et al. 2013), and (Górecki and Eulenstein 2012)). Another category of methods aims at jointly inferring gene trees and species trees (eg. PHYLDOG (Boussau et al. 2013), DLRS (Sjöstrand et al. 2012), PhyloNet (Y. Wang and Nakhleh 2018), BP&P (Rannala and Yang 2017)). The later provide accurate reconstructions by modeling and resolving the discordance between gene trees and species trees (Maddison 1997), but they are also limited by a scalability problem for larger data sets. Despite the improvement in the accuracy and resolution of gene trees reconstructed by all these approaches, there is still a large room for improvement on the accuracy of gene trees and the computing time.

In this work, we approach the question of improving the quality of multiple sequence alignments used in phylogenomics methods for gene tree reconstruction, by choosing wisely the set of proteins representing genes of a gene family. A key factor for the accuracy of multiple sequence alignment is that aligned sequences should be homologs. It is well known that alternative splicing (AS) is a ubiquitous process in eukaryote organisms by which multiple transcripts and proteins are produced from a single gene (Keren, Lev-Maor, and Ast 2010; Nilsen and Graveley 2010; Blencowe 2006; Blencowe 2017; Modrek and Lee 2002; Kuitche, Lafond, and Ouangraoua 2017; Iñiguez and Hernández 2017). Gene tree reconstruction methods account for this information, by selecting and making use of one reference transcript/protein sequence per gene to reconstruct the gene trees that populate gene tree databases (eg. EnsemblCompra (Vilella et al. 2009), PhylomeDB (Huerta-Cepas et al. 2014), Panther (Mi, Muruganujan, and Thomas 2012), TreeFam (Schreiber et al. 2014)). Among existing gene tree databases, the EnsemblCompara method is the most popular one (Vilella et al. 2009). Like most of the current gene tree reconstruction methods, the EnsemblCompara method chooses the longest transcript for each gene as its representative transcript to

generate a multiple sequence alignment. This choice ignores the transcripts of all the other genes of the gene family. The rationale of this choice is that the longest transcript is the more representative because its coverage of the gene is the largest (Kasukawa et al. 2004). However, there is no guarantee that the set of representative transcripts composed of the longest transcript of each gene constitutes the set of the most homologous transcripts between genes. The homology level of two transcripts from two homologous genes can be measured by the quantity of homologous nucleotides and exons shared by the transcripts. The longest transcripts of two homologous genes may be composed of completely different exons. Thus, choosing the longest transcript for each gene, independently of other genes, may lead to poorly homologous transcripts, and then erroneous sequence alignments and gene trees. One of the main consequences of choosing a set of non homologous transcripts is that the alignment of those sequences will forcibly align some of those sequences together,which will muddle the phylogenetic signal. Ideally, the set of transcripts chosen to estimate a gene tree must be equivalogs (Haft et al. 2001). Equivalogs are homologous transcripts that have conserved their functions since their last common ancestor. Equivalogs share the same functions, as well as conserved sequences and homologous exons, which results in similar splicing structures.

Classical methods for clustering homologous transcripts into equivalog families and automated functional identification of proteins rely on sequence and folding structure similarity. This approach ignores the splicing structure similarity between transcripts (Haft et al. 2001; Pandit et al. 2002; Zhang and Skolnick 2005; Balamurugan, Dekker, and Waldmann 2005; Balaji and Srinivasan 2007; Xu and Zhang 2010). However, it has been shown that the splicing structure of homologous sequences is often more conserved than the nucleotide sequences (Betts et al. 2001; Abril, Castelo, and Guigó 2005; Poulos et al. 2011). It has also been shown that the splicing structure contains information that can improve protein and transcript comparison (Abascal, Tress, and Valencia 2015; Abascal, Ezkurdia, et al. 2015). IsoSel (Philippon et al. 2017) is a tool devoted to the selection of equivalogs in the context of phylogenetic reconstruction. It provides a better alternative of choosing the longest isoforms per gene. It uses multiple sequence alignment and bootstrapping to identify equivalogs.

In this paper, we introduce a new framework for the reconstruction of eukaryote gene trees. We consider all transcripts of all genes of a gene family, and we define a new similarity measure by combining splicing structure-based similarity and sequence-based similarity. Based on the all-against-all similarity matrix between transcripts, we propose a clustering method, based on an improved version of the fuzzy c-means (FCM) algorithm (Bezdek, Ehrlich, and Full 1984) and a latent representation of transcripts. This method allows to obtain splicing homology groups, that represent estimated equivalog families. Finally, we use a multiple sequence alignment of the selected representative transcripts to build the gene tree.

The paper is organized as follows: the next section describes the method devised to estimate gene trees by taking into account the splicing structure of

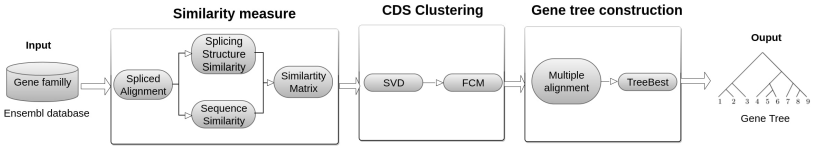


Fig. 1: **Overall view of the method**: The first step consists in the computation of the similarity matrix between Coding DNA Sequence (CDS) of transcripts. The second step consists in grouping CDS into splicing homology groups and, finally, the last step consists in aligning the representative CDS and building the gene tree.

transcripts. The Results and Discussion section presents the results of a comparative analysis between the gene trees estimated using the new method, IsoSel, and, the EnsemblCompara gene trees which are currently the most largely used. The trees reconstructed with our method compare favorably with the trees from the EnsemblCompara database, in terms of the significance of a likelihood difference computed using an Approximately unbiased test (Shimodaira 2002), and in terms of the reconciliation with a species tree.

2 Materials and Methods

We first introduce some formal definitions that will be useful for the description of the method. Next, we present the definition of the new measure of similarity between transcripts, and then the new algorithm for clustering transcripts from a similarity matrix. In the last part of this section, we present how the gene trees are estimated from a set of representative transcripts. An overview of the method is depicted in Figure 1.

2.1 Preliminary definitions

Gene and CDS: A gene sequence is a DNA sequence on the alphabet of nucleotides $\sigma = \{A, C, G, T\}$. Given a gene sequence g, an exon of g is a transcribed segment of g. A coding DNA sequence (CDS) of g is a concatenation of translated exons of g. The following is an example of toy gene g with two alternative CDS c_1 and c_2. The alignment of g , c_1 and c_2 shows that c_1 is composed of 3 exons, and c_2 of 4 exons. Here, CDS refers to transcript or isoform.

```
>g
CGTATGGAATGCGTAAGAAGCAGGTCGTAAGCATACGTGGGTAAGGGGAATGATTGAAAG
>c1
ATGCAAGCAGGTCGGGAATGA
>c2
ATGGAATGCAAGCAGCATACGTGGGGGAATGATTGA
```

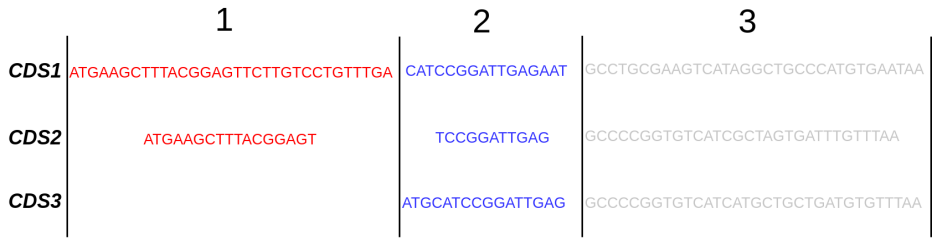


Fig. 2: **Multiple splicing structure**: Multiple splicing structure of three homologous CDS decomposed into three segment classes represented by different colors: red color for the first class, blue for the second class, and grey for the third class. Each CDS has a non-empty segment in all segment classes, except CDS3 which has an empty segment in the first class.

```
g : CGTATGGAATGCGTAAGAAGCAGGTCGTAAGCATACGTGGGTAAGGGGAATGATTGAAAG
c1: --------ATGC-----AAGCAGGTC-------------------GGGAATGA
c2: ---ATGGAATGC-----AAGCAG--------CATACGTGG-----GGGAATGATTGA
```

Multiple splicing structure: Given a set of CDS $\mathcal{C}$ from a set of homologous genes, a segment class of $\mathcal{C}$ is a set composed of exactly one, possibly empty, segment from each element of $\mathcal{C}$, such that these segments are homologous, i.e descending from the same ancestral segment. A segment class must contain at least one non-empty segment. A multiple splicing structure of a set of CDS $\mathcal{C}$ is a chain $E = \{E[1], ..., E[n]\}$ of segment classes of $\mathcal{C}$ in which no pair of classes $E[i]$ and $E[j]$ contains an overlapping segment, and each CDS c in $\mathcal{C}$ is the concatenation of its segments belonging to classes of E. Figure 2 presents an example of multiple spliced structure of three CDS decomposed into three segment classes. In practice, a set of homologous CDS is decomposed into a chain of segment classes corresponding to homologous exon classes.

A multiple spliced alignment : Given a set of homologous CDS $\mathcal{C}$ decomposed into a multiple splicing structure $E = \{E[1], ..., E[n]\}$, the multiple spliced alignment of $\mathcal{C}$ corresponding to E is a chain $A = \{A[1], ..., A[n]\}$ such that each $A[i]$ is an alignment of the segment class $E[i]$. For any CDS c in $\mathcal{C}$, $A_c = \{A[i]_{|c}|1 \leq i \leq n\}$ is the chain of n gapped segments of c induced by the multiple spliced alignment A. Given a multiple spliced alignment A, $|A|$ denotes the number of sub-alignments composing A, i.e. the number of segment classes in the corresponding multiple spliced alignment. Given a gapped nucleotide segment x, $len(x)$ denotes the number of nucleotides in x. Figure 3 gives an example of multiple spliced alignment corresponding to the multiple splicing structure of three CDS from Figure 2.

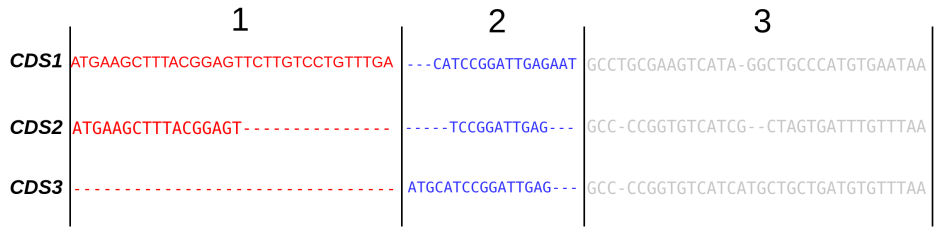


Fig. 3: **Multiple spliced alignment**: a multiple spliced alignment A corresponding to the multiple splicing structure of three CDS from Figure 2. For instance, $|A| = 3$, $len(A[1]_{|CDS3}) = 0$, and $len(A[2]_{|CDS3}) = 11$.

2.2 Similarity measure

The similarity measure between two CDS, x and y, of a set of homologous CDS is defined as follows:

$$Sim(x, y) = \alpha \times Struct(x, y) + (1 - \alpha) \times Seq(x, y)$$

$Struct(x, y)$ is a splicing structure similarity between x and y, $Seq(x, y)$ is a sequence similarity between x and y, and α is such that $0 \leq \alpha \leq 1$ and defines the relative weights of $Struct(x, y)$ and $Seq(x, y)$ in the similarity score.

The value of $Struct(x, y)$ is defined by combining three similarity scores:

$$Struct(x, y) = \frac{SE(x, y) + TP(x, y) + SL(x, y)}{3}$$

$SE(x, y)$, $TP(x, y)$ and $SL(x, y)$ are respectively a splicing event (SE) similarity score, a translation phase (TP) similarity score, and a segment length (SL) similarity score between x and y, defined thereafter. Here, translation phase refers to the reading frame. It may take the values 0, 1 or 2.

Splicing event similarity: Given a multiple spliced alignment A of a set of CDS $\mathcal{C}$, we consider four possible types of alternative splicing events for each gapped segment $A[i]_{|x}$ such that $1 \leq i \leq |A|$ and $x \in \mathcal{C}$. Figure 4 illustrates the different types of alternative splicing events. An exon skipping (ES) arises when the segment contains only gaps. For instance, the first segment of CDS3 in Figure 4 has an ES event. A 3' alternative splice site (3P) event arises when the segment starts with gaps. For instance, the second segments of CDS2 and CDS3 have a 3P event. A 5' alternative splice site (5P) event arises when the segment ends with gaps. For instance, the first segment of CDS2 as well as the second segments of CDS1 and CDS3 have a 5P event. An internal gap (IG) event arises when there is a set of consecutive gaps within the segment. For instance, the third segment of CDS1, CDS2 and CDS3 have at least one IG event.

For a CDS x, we denote by $e[i]_{|x}$ the set of alternative splicing events displayed by the gapped segment $A[i]_{|x}$. Given two CDS x and y from $\mathcal{C}$, the splicing

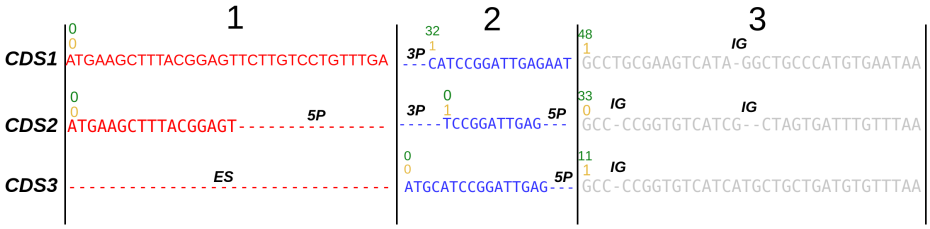


Fig. 4: **Alternative splicing events**: The multiple spliced alignment of Figure 3 with gapped segments annotated with their alternative splicing events, translation phases, and segment lengths. Above the first nucleotide of each segment, the number in green represents the position of the first nucleotide in the CDS and the number in orange represents the translation phase of the exon. The alternative splicing events of each segment are indicated (Exon Skipping(ES), 3 Prime Extension(3P), 5 Prime Extension(5P), or Internal Gap(IG)).

event similarity between the two gapped segments $A[i]_{|x}$ and $A[i]_{|y}$ is the ratio of the number of common events in the two sets $e[i]_{|x}$ and $e[i]_{|y}$:
$SE(A[i]_{|x}, A[i]_{|y}) = \frac{|e[i]_{|x} \cap e[i]_{|y}|}{|e[i]_{|x} \cup e[i]_{|y}|}$
The splicing event similarity between x and y is then given by:
$SE(x,y) = \sum_{i=1}^{i=|A|} \frac{SE(A[i]_{|x}, A[i]_{|y})}{|A|}$

Translation phase similarity: Given a multiple spliced alignment A of a set of CDS $\mathcal{C}$, for each gapped segment $A[i]_{|x}$ such that $1 \leq i \leq |A|$ and $x \in \mathcal{C}$, we denote by $phase[i]_{|x}$ the translation phase of the segment $A[i]_{|x}$. The value of $phase[i]_{|x}$ equals the position of the first nucleotide of $A[i]_{|x}$ in the sequence x modulo 3. Given two CDS x and y from $\mathcal{C}$, the translation phase similarity between the two gapped segments $A[i]_{|x}$ and $A[i]_{|y}$ is 1 if the two segments have the same translation phase, and 0 otherwise:
$TP(A[i]_{|x}, A[i]_{|y}) = \begin{cases} 1 \text{ if } phase[i]_{|x} = phase[i]_{|y} \\ 0 \text{ otherwise.} \end{cases}$
The translation phase similarity between x and y is then given by:
$TP(x,y) = \sum_{i=1}^{i=|A|} \frac{TP(A[i]_{|x}, A[i]_{|y})}{|A|}$

Segment length similarity: Given two CDS x and y from $\mathcal{C}$, the segment length similarity between the two gapped segments $A[i]_{|x}$ and $A[i]_{|y}$ is given by:
$SL(A[i]_{|x}, A[i]_{|y}) = 1 - \frac{|len(A[i]_{|x}) - len(A[i]_{|y})|}{len(A[i]_{|x}) + len(A[i]_{|y})}$
The segment length similarity between x and y is then given by:
$SL(x,y) = \sum_{i=1}^{i=|A|} \frac{SL(A[i]_{|x}, A[i]_{|y})}{|A|}$

Sequence similarity: Giving a multiple spliced alignment A of a set of CDS $\mathcal{C}$, we use the substitution matrix *BLOSUM100* as substitution model to compute the sequence similarity score between each pair of CDS in $\mathcal{C}$.

Based on the definition of the similarity score $Sim(x, y)$ between two CDS x and y, an all-against-all similarity matrix is computed for the set of homologous CDS $\mathcal{C}$ of any gene family.

2.3 CDS Clustering

In order to determine the representative CDS of a set of homologous genes, CDS are first clustered into splicing homology groups. It has been demonstrated that, for clustering problems where clusters may overlap each other, fuzzy clustering methods, which can assign a data to more than one cluster, achieve higher performance than hard clustering methods, which assign each data to a single cluster (Sun, S. Wang, and Jiang 2004). The CDS clustering problem is best approach as a fuzzy clustering problem because a CDS in one gene may be similar in sequence and splicing structure to several alternative CDS of another gene, and splicing homology groups could overlap.

The Fuzzy C-means (FCM) algorithm and its derivatives are the most widely used fuzzy clustering algorithm (Bezdek, Ehrlich, and Full 1984). As input, they require a representation of the data, generally a vectorial representation, which facilitates computing of the centroid of a cluster of data. They also require as input the number of clusters in the given data set. The clustering method starts with a transformation of the CDS representation into a vectorial representation, and the application of the FCM-Based Splitting Algorithm (FBSA) (Sun, S. Wang, and Jiang 2004) to cluster CDS.

Vectorial representation of CDS: In this section, we propose to compute a latent vectorial representation of CDS from a $N \times N$ similarity matrix S, containing the pairwise similarity scores between the N CDS of a set of homologous genes. Such a representation allows CDS to be represented in a new geometric space while preserving their mutual similarity and facilitating computing of centroids in clustering algorithms. Each CDS will be represented as an N-dimensional vector by applying a spectral decomposition on the similarity matrix S. The spectral decomposition of a symmetric $N \times N$ matrix S is the decomposition of S into $S = UDU^T$, such that U is an orthogonal $N \times N$ matrix and D is a $N \times N$ diagonal matrix. The columns of U correspond to the eigen vectors of S, and the diagonal entries of D correspond to the eigen values.

We define an $N \times N$ matrix $X = U \times \sqrt{D}$, such that each row x_i of X is the new vectorial representation of the i^{th} CDS from the similarity matrix S. This transformation is valid because the similarity score S_{ij} between the i^{th} and j^{th} CDS is now approximately equal to the inner vector product $< x_i, x_j >$ of the vectors representing the CDS. Thus, the relative similarity scores between CDS are preserved by the transformation, and the vectors $\{x_1, \ldots, x_N\}$ can be clustered using a fuzzy clustering algorithm.

Application of the FBSA for fuzzy clustering:

The FCM algorithm requires, as input parameter, the number of clusters in the data-set in order to be clustered (Bezdek, Ehrlich, and Full 1984). Given N data vectors $X = \{x_1, x_2, ..., x_N\}$ to cluster, and c the expected number of

clusters, the FCM algorithm searches for a $N \times c$ non-negative matrix M called the fuzzy partition matrix. Each value m_{ki} such that $1 \leq k \leq N$ and $1 \leq i \leq c$ in the matrix M is the membership value of vector x_k to the i^{th} cluster, and for any vector x_k, $\sum_{i=1}^{c} m_{ki} = 1$. Given a $N \times c$ membership matrix M, the centre of each cluster i, $1 \leq i \leq c$, is computed as the vector:

$$v_i = \frac{\sum_{k=1}^{N} m_{ki}^f x_k}{\sum_{k=1}^{N} m_{ki}^f} 1$$

The exponent $f > 1$ in the cluster centre formula is a parameter called a fuzzifier. Given a set of cluster centres $V = \{v_1, \ldots, v_c\}$, the values of the $N \times c$ membership matrix M are computed as follows:

$$m_{ki} = \begin{cases} (\sum_{j=1}^{c} (\frac{\|x_k - v_i\|}{\|x_k - v_j\|})^{\frac{2}{f-1}}) & if \, \|x_k - v_i\| > 0, \forall j \\ 1 & if \, \|x_k - v_i\| = 0 \\ 0 & if \exists j \neq i \, \|x_k - v_j\| = 0 \end{cases} \quad 2$$

The formula (1) and (2) are derived from minimization of the following objective function F_f.

Given N data vectors $X = \{x_1, x_2, ..., x_N\}$ and an expected number of clusters c, the FCM algorithm computes a membership matrix M and a set of cluster centres $V = \{v_1, \ldots, v_c\}$ that minimizes the following function

$$F_f(U, V) = \sum_{k=1}^{N} \sum_{i=1}^{c} m_{ki}^f \|x_k - v_i\|^2 \, 3$$

The FCM algorithm starts by randomly initializing the cluster centres $V = \{v_1, \ldots, v_c\}$. Then it iteratively recalculates the membership matrix M using equation (2) and a new set of cluster centres using equation (1) until a set of stable cluster centres is reached.

The main limitation of the FCM algorithm is that it requires the number of clusters as input. Several derivatives of the FCM algorithm have been developed for automatically determining the number of clusters. As input parameters, they take a lower bound c_{min} and an upper bound c_{max} for the number of clusters, compute an optimal clustering using FCM for each number of cluster c such that $c_{min} \leq c \leq c_{max}$, and choose the best value of c based on a cluster validity criterion.

The FBSA algorithm is a derivative of the FCM algorithm that allows to cluster a data set while automatically determining the number of clusters (Sun, S. Wang, and Jiang 2004). FBSA improves on other derivatives of the FCM algorithm to allows automated selections of the number of clusters. It reduces the randomness in the initialization of cluster centres at the beginning of each clustering phase. For each clustering phase with a value of c such that $c_{min} < c \leq c_{max}$, the clustering is carried out starting with the previously obtained $c-1$ clusters and splitting the worst cluster in two clusters to obtain c clusters. The

worst cluster is the one corresponding to the minimum value of a score function $S(i)$ associated with each cluster i as follows:

$$S(i) = \frac{\sum_{k=1}^{N} m_{ki}}{number\_of\_data\_vectors\_in\_cluster\_i}$$

A second improvement of FBSA in the selection of the best number of clusters is the use of a new validity index $V_d(c)$ based on a linear combination of cluster compactness and separation that is efficient even when clusters overlap each other. The validity index $V_d(c)$ is computed for each cluster number c to choose the number of clusters that maximizes $V_d(c)$. The detailed description of FBSA and the definition of the validity index are provided in (Sun, S. Wang, and Jiang 2004).

Given a $N \times N$ similarity matrix S on a set of N homologous CDS $\mathcal{C}$ of a gene family, we extend the FBSA method to cluster $\mathcal{C}$ into subsets of splicing homology groups, and select a set of representative CDS as follows:

1. Apply a spectral decomposition on the matrix S to obtain a vectorial representation $X = \{x_1, \ldots, x_N\}$ of the N CDS in $\mathcal{C}$;
2. Apply FBSA on $X = \{x_1, \ldots, x_N\}$ and choose the best clustering corresponding to a number c_{opt} of clusters maximizing $V_d(c_{opt})$;
3. Select the cluster i in the c_{opt}-clustering such that $S(i)$ is maximal.
4. For each gene g, select as representative CDS, the CDS k of g such that the membership M_{ki} of x_k to the ith cluster is maximum.

2.4 Gene tree construction

In order to carry out a comparative study on the effect of the choice of representative CDS for gene tree construction, the new method and IsoSel method use the same steps as the Ensembl method, except for the representative CDS selection step. After selecting the representative CDS, the third methods compute a multiple sequence alignment of the protein sequences corresponding to the representative CDS using Mafft (Katoh and Standley 2013). Next, the CDS back-translated protein alignment and the species tree are used to estimate the gene trees with the TreeBest phylogenetic reconstruction method. TreeBest is a method for building gene trees using a known species tree (Schreiber et al. 2014).

3 Results and Discussion

In order to evaluate the new gene tree reconstruction method, we compare it with the method from EnsemblCompara and the method from IsoSel, based on a dataset of gene families from the EnsemblCompara database (Vilella et al. 2009). In the remaining section, since the differences between the three compared methods lies within the approach used to select a set of representative transcripts

for genes, the methods are named according to the latter. The EnsemblCompara method that selects the longest transcript of each gene is named the Longest Transcript (`LT`) method, The IsoSel method that select a set of isoform transcript is name Isosel, and the new method that computes and uses a splicing homology group is called the Splicing Homology Transcript (`SHT`) method.

3.1 Dataset and compared methods

The initial dataset contains 2036 gene families of 173 species selected randomly from the EnsemblCompara database (release 98) (ibid.). The data for each gene family is composed of a set of homologous genes, their Coding Dna Sequences (CDS) and the gene tree from the EnsemblCompara database called the `LT_db` gene tree. For each gene, only CDS that start with a start codon and whose lengths are multiple of 3 are considered, because the information on the coding phase is required for computing the similarity scores between CDS in the first step of the method, and the translations into protein sequences are needed for building the gene tree using TreeBest (Schreiber et al. 2014) in the last step of the method. Thus, the LT_db gene tree contains only genes for which at least one CDS is considered. The LT_db gene tree is induced from the EnsemblCompara database gene tree that is often on a larger set of genes. IsoSel gene tree is the tree obtained by applying TreeBest on the transcript isoforms return by IsoSel.

Figure 5 shows some statistics on the number of genes per family and the number of CDS per gene in the dataset. Figure 5 (a) shows the percentage of gene families per ratio of the number of genes retained in this study over the number of genes in the EnsemblCompara database. For instance, we observe that in more than 45% of gene families, the number of genes used in this study is less than 10% of the number of genes in the EnsemblCompara gene family. Figure 5 (b) shows the percentage of genes per number of CDS. We observe that almost 80% of genes have only one CDS, and 15% of genes have two CDS. Figure 5 (c) shows the percentage of gene families per percentage of genes familly having a single CDS. We note that more than 60% of gene families have more than 90% of genes having a single CDS. Out of 2036 gene families, 1313 families have a single CDS per gene, which induces a single possible set of representative transcripts, thus the same tree is reconstructed using the two methods. These families were discarded, leaving a dataset with 723 gene families for the comparison. We call this dataset the 723-dataset, and the initial dataset is called the 2036-dataset. Figure 5 (d) and (e) show the statistics on the number of CDS per gene in the 723-dataset. For instance, we observe that the number of single-CDS genes in the dataset is still high (more than 55 %), but the percentage of gene families that have more than 90% single-CDS genes is lower (less than 5% of gene families).

Despite the large number of single-CDS genes in the dataset, the comparative analyses between the LT, IsoSel and SHT reveal significant differences between the results of the two methods, as shown in the sequel.

The LT and IsoSel methods were applied on the gene families to reconstruct a set of gene trees respectively called the `LT` gene trees and IsoSel gene trees. Note

that, for the same gene family, the LT gene tree may differ from the LT_db gene tree, because the LT_db is induced from the EnsemblCompara database gene tree that was often constructed on a larger set of genes as shown in Figure 5 (a). Using the SHT method, the gene trees were reconstructed for 7 different values of the parameter α of the SHT method, $\alpha = 0.0, 0.2, 0.4, 0.5, 0.6, 0.8, 1.0$, which corresponds to the weight of the splicing-structure similarity in the similarity scores between CDS. This yielded 7 sets of gene trees for the SHT method called the `SHT` gene trees. For each value of α, the resulting SHT gene trees were compared with the corresponding LT, LT_db and IsoSel gene trees.

3.2 Comparison criteria

Four criteria are used to compare the SHT, IsoSel and LT gene trees for each value of $\alpha = 0.0, 0.2, 0.4, 0.5, 0.6, 0.8, 1.0$. The first criterion is the percentage of common transcripts between the sets of representative transcripts computed in the LT and SHT method, and also between the sets of representative transcripts computed in the IsoSel and SHT method. This criterion is used to evaluate the similarity between the methods in terms of the set of selected representative transcripts. The second criterion is the percentage of trees from each method which pass an approximately unbiased (AU) test (Shimodaira 2002) used to evaluate the significance of a likelihood difference between candidate trees on the same set of taxa. The third criterion is the comparison of the reconciliation cost of the gene trees with the species trees. The last criterion is the comparison of species deduced from the set of gene trees built by each approach.

3.3 Comparison based on the percentage of common representative transcripts

Figure 6 shows the percentage of common representative transcripts between the LT and SHT methods. We observe that the representative transcripts chosen by the SHT method are not always the longest ones. Figure 6 (a) shows that for various values of α, the median percentage of common representative transcripts in gene families is almost 70% for the 723-dataset. Figure 6 (b) shows that when we consider only genes having more than one transcript, the median percentage of common representative transcripts decreases and it is almost 37%. Figure 6 (c) shows the average ratio of the mean similarity between the SHT transcripts over the mean similarity between the LT transcripts. As expected, we observe that the SHT transcripts are closer in terms of similarity than the LT transcripts. These results illustrate that the SHT method is effective at computing a set of representative transcripts different from the longest transcripts selected by the LT. The SHT representative transcripts are closer in terms of similarity than the LT transcripts are when we increase the weight α of the splicing structure similarity score.

We also observe very few changes between the results of various values of α that determine the weights of the splicing structure similarity and the sequence

similarity scores in the CDS similarity measure. It suggests that there is a correlation between the splicing structure similarity and the sequence similarity scores. This is confirmed by the mean of the correlation scores between the two scores computed for all gene families of the 2036-dataset: minimum correlation score equals 0.22, maximum 1.0, mean 0.64, and median 0.57.

Figure 7 shows similar results when comparing the percentage of common representative transcripts between IsoSel and SHT methods. Figure 7 a) shows that for various values α, the median percentage of common representative transcripts in gene families is almost 70% for the 723-dataset. Figure 6 (b) shows that when we consider only genes having more that one transcript, the median percentage of common representative transcripts is almost 30%. Figure 7 c) shows that the SHT representative transcripts are closer in terms of similarity than the IsoSel representative transcripts.

3.4 Comparison based on the percentage of trees passing the AU test

We used the approximately unbiased (AU) test to compare the trees built by the LT and the SHT methods (ibid.). The AU test compares the likelihood of a set of trees in order to build a confidence set, which will consists of the trees that cannot be statistically rejected as a valid hypothesis. We say a tree can be rejected if its AU value, interpreted as a $p - value$, is under 0.05. Otherwise, no significant evidence allows us to reject one of the two trees. We start by executing PhyML (release 27) (Guindon and Gascuel 2003) on the trees to obtain the log-likelihood values per site and we execute Consel (release 28) to obtain the results from the AU test.

Figure 8 shows the percentage of LT_db, LT, IsoSel and SHT trees passing the AU test for various values of α. Since the AU test requires gene sequences, the sequences of transcripts selected by Ensembl, IsoSel and SHT have been used as gene sequences. Figures 8 a), b), and, c) respectively illustrates the results obtained when the gene sequences are the transcript sequences selected by SHT, LT, and, IsoSel. When the transcript selected by SHT are used, we clearly observed that the SHT method slightly better than LT_db, LT and the IsoSel method. Between 78% to 79% of the SHT gene trees pass the AU test. 79% of trees passing the AU test are obtained for $\alpha = 0.8$. Whereas, for the LT_db trees, this percentage varies between 68% and 70%, for LT trees, this percentage varies between 74% and 75%, and for IsoSel trees between 76% and 77%. When transcripts selected by LT Figure 8 b) or IsoSel 8 C) are used, we observe that each both methods performs also slightly better. This can be explain by the fact the transcript sequences choose for AU test favors the the tree having the transcript better than other trees. For this criteria, the results clearly illustrate that the gene trees from the fourth approaches are all comparable based on the results of AU test. The SHT and IsoSel gene trees also allow to identify cases in which the LT gene trees do not pass the AU test, this confirms that the choice of the longest transcript is not always the best approach to select a set of representative transcripts. The results also allow to compare the accuracy of the

LT and LT_db gene trees. They show that using a larger set of genes provides higher accuracy for the LT_db gene trees as compared to the LT trees that are computed on a restricted set of genes.

3.5 Comparison based on the reconciliation cost with a species tree

Figure 9 (a) shows the comparative results between the SHT, and, LT gene trees in terms of reconciliation cost on the 723-dataset. We observe that, while 65% of gene tree build by SHT and LT are eqauls, 20% are differents but have the same reconciliation cost. For the remaining gene trees, SHT always have a highest percentage of tree having the lowest reconciliation cost when varying the values of α. This percentage varying between 8 to 10%, while the same percentage for LT varying between 6 to 8%. Figure 9 (b) shows the comparative results between the SHT and the LT_db gene trees on the 2036-dataset. Here, 72% of gene tree build by both approaches are equals, while almost 17% are differents but have the same reconciliation cost. For the remaining gene trees, SHT always have the highest percentages of gene trees having the lowest reconciliation cost when varying the values of α Figure 9 (c) shows the comparative results between the SHT and the IsoSel gene trees on the 2036-dataset. Here, the percentages of equal gene trees build by SHT and IsoSel is almost 87%. While the percentage of differents gene trees having the same reconciliation cost varying between 8 to 10%. For the remaining gene tree, there is remarkable difference, while only 1% of gene tree build by IsoSel have a lowest reconciliation cost, 3 to 4% of gene trees build by SHT have lowest reconciliation cost. This results supports the use of the new similarity measure and clustering method to select sets of representative transcripts more wisely.

This result, coupled with the results of Figure 6 and 7 on the fourth methods' accuracy comparison, (LT < LT_db < IsoSel < SHT) shows that choosing wisely the set of representative transcripts (i.e SHT trees) allows to reach the accuracy obtained using a larger set of genes, and even to compute more accurate gene trees.

3.6 Comparison based on the evaluation of the deduced species trees

The set of gene trees built by LT, LT_db, IsoSel, and, SHT ($\alpha = \{0.0, 0.2, 0.4, 0.5, 0.6, 0.8, 1.0\}$) were used as input of Astral (Mirarab and Warnow 2015) to build a species tree. 5 percentages (20%, 40%, 60%, 80%, and, 100%) of gene trees were used as input of Astral. We used criteria two criteria to compare theses species trees. The first criterion is the quartet support values of branches of each species. The quartet score is the proportion of input gene tree quartet trees satisfied by the species tree. This is a number between zero and one; the higher this number, the less discordant your gene trees are. Figure 10 shows the comparison results between LT, LT_db, IsoSel, and the 7 species trees obtained each value of α. We note on this boxplot that the quartet support values of each method increase with the percentage of gene trees used. When 20%, 40,%, 60% or 80% of gene trees

are used, the quartet support values of methods are slightly equals. When 100% of gene trees are used, the quartet support values of SHT is higher compare to one of LT, LT_db, and, SHT. Figure 11 shows the percentages of common clades between reconstruct species trees when varying the percentage of gene trees used and the initial Ensembl species tree. We observe that SHT has the highest percentage of common clades with the species tree.

4 Conclusion

A new approach called Splicing Homology Transcript (SHT) method has been devised to estimate eukaryote gene trees. It is based on a novel and alternative way to choose the representative transcripts of a set of homologous genes. Contrary to the most used approaches that select the longest transcript of each gene as the representative transcript independently of other homologous genes, it selects a set of representative transcripts using a combination of the sequence similarity and the splicing structure similarity between sequences. This approach yields more accurate gene trees that the approach that selects the longest transcript of each gene as the representative transcript. We predict that alternative-splicing-aware gene tree reconstruction methods will be instrumental to advancing our understanding of gene family and genome evolution.

The analysis of the sequence and the structure similarity scores between transcripts of a gene family shows that the two scores are correlated. In this study, we have enriched the similarity measure definition based on the sequence by including information related to the structure, it will be interesting as a future work to carry out machine learning in order to determine the weight allowing to obtain better results. As long as this study exploit the spliced structure of sequences, the use of well-annotated species may increase the results accuracy. Moreover, the SHT method differs from the EnsemblCompara and IsoSel gene tree estimation method only by the strategy used in the representative transcripts's selection step. Future works will consider alternative strategies in the other steps, such as the multiple sequence alignment and the phylogeny estimation steps. The method also selects a single set of representative transcripts while there can be several co-optimal sets of representative transcripts. Future extensions of the method will include the computation of a transcript tree for each co-optimal set of representative transcripts, and the combination of the resulting trees to increase the precision of the final gene tree. Finally, a further extension of this work will consist of combining the set of representative transcript trees into a transcript phylogeny representing the evolutionary history of all transcripts of a set of homologous genes.

5 Acknowledgements

The authors thank the CoBiUS lab at University of Sherbrooke for their helpful constructive discussions.

References

Abascal, Federico, Iakes Ezkurdia, et al. (2015). "Alternatively spliced homologous exons have ancient origins and are highly expressed at the protein level". In: *PLoS computational biology* 11.6, e1004325.

Abascal, Federico, Michael L Tress, and Alfonso Valencia (2015). "The evolutionary fate of alternatively spliced homologous exons after gene duplication". In: *Genome biology and evolution* 7.6, pp. 1392–1403.

Abril, Josep F, Robert Castelo, and Roderic Guigó (2005). "Comparison of splice sites in mammals and chicken". In: *Genome research* 15.1, pp. 111–119.

Åkerborg, Örjan et al. (2009). "Simultaneous Bayesian gene tree reconstruction and reconciliation analysis". In: *Proceedings of the National Academy of Sciences* 106.14, pp. 5714–5719.

Balaji, S and N Srinivasan (2007). "Comparison of sequence-based and structure-based phylogenetic trees of homologous proteins: Inferences on protein evolution". In: *Journal of biosciences* 32.1, pp. 83–96.

Balamurugan, Rengarajan, Frank J Dekker, and Herbert Waldmann (2005). "Design of compound libraries based on natural product scaffolds and protein structure similarity clustering (PSSC)". In: *Molecular BioSystems* 1.1, pp. 36–45.

Bansal, Mukul S et al. (2015). "Improved gene tree error correction in the presence of horizontal gene transfer". In: *Bioinformatics* 31.8, pp. 1211–1218.

Betts, Matthew J et al. (2001). "Exon structure conservation despite low sequence similarity: a relic of dramatic events in evolution?" In: *The EMBO journal* 20.19, pp. 5354–5360.

Bezdek, James C, Robert Ehrlich, and William Full (1984). "FCM: The fuzzy c-means clustering algorithm". In: *Computers & Geosciences* 10.2-3, pp. 191–203.

Blencowe, Benjamin J (2006). "Alternative splicing: new insights from global analyses". In: *Cell* 126.1, pp. 37–47.

— (2017). "The relationship between alternative splicing and proteomic complexity". In: *Trends in biochemical sciences* 42.6, pp. 407–408.

Boussau, Bastien et al. (2013). "Genome-scale coestimation of species and gene trees". In: *Genome research* 23.2, pp. 323–330.

Chen, Kevin, Dannie Durand, and Martin Farach-Colton (2000). "Notung: dating gene duplications using gene family trees". In: *Proceedings of the fourth annual international conference on Computational molecular biology*, pp. 96–106.

Górecki, Pawel and Oliver Eulenstein (2012). "Algorithms: simultaneous error-correction and rooting for gene tree reconciliation and the gene duplication problem". In: *BMC bioinformatics*. Vol. 13. S10. Springer, S14.

Guindon, Stéphane and Olivier Gascuel (2003). "A simple, fast, and accurate algorithm to estimate large phylogenies by maximum likelihood". In: *Systematic biology* 52.5, pp. 696–704.

Haft, Daniel H et al. (2001). "TIGRFAMs: a protein family resource for the functional identification of proteins". In: *Nucleic acids research* 29.1, pp. 41–43.

Hahn, Matthew W (2007). "Bias in phylogenetic tree reconciliation methods: implications for vertebrate genome evolution". In: *Genome biology* 8.7, R141.

Huerta-Cepas, Jaime et al. (2014). "PhylomeDB v4: zooming into the plurality of evolutionary histories of a genome". In: *Nucleic acids research* 42.D1, pp. D897–D902.

Iñiguez, Luis P and Georgina Hernández (2017). "The evolutionary relationship between alternative splicing and gene duplication". In: *Frontiers in genetics* 8, p. 14.

Kasukawa, Takeya et al. (2004). "Construction of representative transcript and protein sets of human, mouse, and rat as a platform for their transcriptome and proteome analysis". In: *Genomics* 84.6, pp. 913–921.

Katoh, Kazutaka and Daron M Standley (2013). "MAFFT multiple sequence alignment software version 7: improvements in performance and usability". In: *Molecular biology and evolution* 30.4, pp. 772–780.

Keren, Hadas, Galit Lev-Maor, and Gil Ast (2010). "Alternative splicing and evolution: diversification, exon definition and function". In: *Nature Reviews Genetics* 11.5, p. 345.

Kuitche, Esaie, Manuel Lafond, and Aıda Ouangraoua (2017). "Reconstructing protein and gene phylogenies using reconciliation and soft-clustering". In: *Journal of bioinformatics and computational biology* 15.06, p. 1740007.

Lafond, Manuel et al. (2013). "Gene tree correction guided by orthology". In: *BMC bioinformatics*. Vol. 14. S15. Springer, S5.

Lartillot, Nicolas and Hervé Philippe (2004). "A Bayesian mixture model for across-site heterogeneities in the amino-acid replacement process". In: *Molecular biology and evolution* 21.6, pp. 1095–1109.

El-Mabrouk, Nadia and Aıda Ouangraoua (2017). "A general framework for gene tree correction based on duplication-loss reconciliation". In: *17th International Workshop on Algorithms in Bioinformatics (WABI 2017)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.

Maddison, Wayne P (1997). "Gene trees in species trees". In: *Systematic biology* 46.3, pp. 523–536.

Mi, Huaiyu, Anushya Muruganujan, and Paul D Thomas (2012). "PANTHER in 2013: modeling the evolution of gene function, and other gene attributes, in the context of phylogenetic trees". In: *Nucleic acids research* 41.D1, pp. D377–D386.

Mirarab, Siavash and Tandy Warnow (2015). "ASTRAL-II: coalescent-based species tree estimation with many hundreds of taxa and thousands of genes". In: *Bioinformatics* 31.12, pp. i44–i52.

Modrek, Barmak and Christopher Lee (2002). "A genomic view of alternative splicing". In: *Nature genetics* 30.1, p. 13.

Nilsen, Timothy W and Brenton R Graveley (2010). "Expansion of the eukaryotic proteome by alternative splicing". In: *Nature* 463.7280, p. 457.

Noutahi, Emmanuel et al. (2016). "Efficient gene tree correction guided by genome evolution". In: *PLoS One* 11.8.

Pandit, Shashi B et al. (2002). "SUPFAM—a database of potential protein superfamily relationships derived by comparing sequence-based and structure-based families: implications for structural genomics and function annotation in genomes". In: *Nucleic acids research* 30.1, pp. 289–293.

Philippon, Héloıse et al. (2017). "IsoSel: Protein Isoform Selector for phylogenetic reconstructions". In: *PloS one* 12.3, e0174250.

Poulos, Michael G et al. (2011). "Developments in RNA splicing and disease". In: *Cold Spring Harbor perspectives in biology* 3.1, a000778.

Rannala, Bruce and Ziheng Yang (2017). "Efficient Bayesian species tree inference under the multispecies coalescent". In: *Systematic biology* 66.5, pp. 823–842.

Rasmussen, Matthew D and Manolis Kellis (2007). "Accurate gene-tree reconstruction by learning gene-and species-specific substitution rates across multiple complete genomes". In: *Genome research* 17.12, pp. 1932–1942.

— (2011). "A Bayesian approach for fast and accurate gene tree reconstruction". In: *Molecular Biology and Evolution* 28.1, pp. 273–290.

Schreiber, Fabian et al. (2014). "TreeFam v9: a new website, more species and orthology-on-the-fly". In: *Nucleic acids research* 42.D1, pp. D922–D925.

Scornavacca, Celine, Edwin Jacox, and Gergely J Szöllősi (2015). "Joint amalgamation of most parsimonious reconciled gene trees". In: *Bioinformatics* 31.6, pp. 841–848.

Shimodaira, Hidetoshi (2002). "An approximately unbiased test of phylogenetic tree selection". In: *Systematic biology* 51.3, pp. 492–508.

Sjöstrand, Joel et al. (2012). "DLRS: gene tree evolution in light of a species tree". In: *Bioinformatics* 28.22, pp. 2994–2995.

Stamatakis, Alexandros (2006). "RAxML-VI-HPC: maximum likelihood-based phylogenetic analyses with thousands of taxa and mixed models". In: *Bioinformatics* 22.21, pp. 2688–2690.

Sun, Haojun, Shengrui Wang, and Qingshan Jiang (2004). "FCM-based model selection algorithms for determining the number of clusters". In: *Pattern recognition* 37.10, pp. 2027–2037.

Szöllősi, Gergely J et al. (2013). "Efficient exploration of the space of reconciled gene trees". In: *Systematic biology* 62.6, pp. 901–912.

Thomas, Paul D (2010). "GIGA: a simple, efficient algorithm for gene tree inference in the genomic age". In: *BMC bioinformatics* 11.1, p. 312.

Vilella, Albert J et al. (2009). "EnsemblCompara GeneTrees: Complete, duplication-aware phylogenetic trees in vertebrates". In: *Genome research* 19.2, pp. 327–335.

Wang, Yaxuan and Luay Nakhleh (2018). "Towards an accurate and efficient heuristic for species/gene tree co-estimation". In: *Bioinformatics* 34.17, pp. i697–i705.

Wu, Yi-Chieh et al. (2013). "TreeFix: statistically informed gene tree error correction using species trees". In: *Systematic Biology* 62.1, pp. 110–120.

Xu, Jinrui and Yang Zhang (2010). "How significant is a protein structure similarity with TM-score= 0.5?" In: *Bioinformatics* 26.7, pp. 889–895.

Zhang, Yang and Jeffrey Skolnick (2005). "TM-align: a protein structure alignment algorithm based on the TM-score". In: *Nucleic acids research* 33.7, pp. 2302–2309.

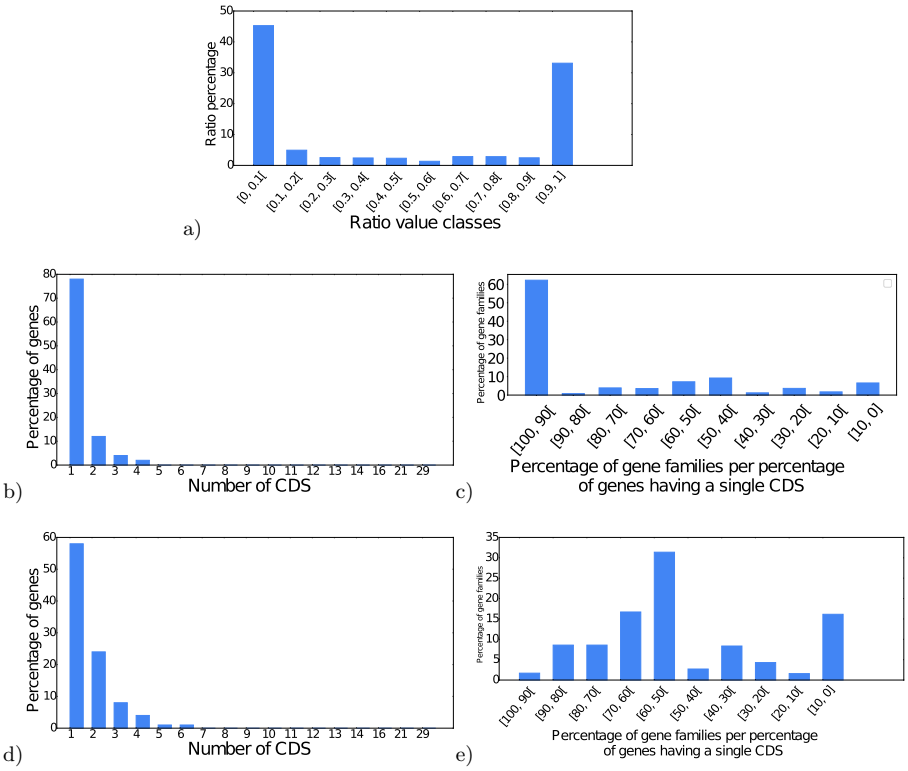


Fig. 5: **Statistics on the number of genes per family and the number of CDS per gene**: **(a)** Percentage of gene families per ratio of the number of genes retained in this study over the number of genes in the EnsemblCompara database. **(b)** Percentage of genes per numbers of CDS in the 2036-dataset. **(c)** Percentage of gene families per percentage of genes having a single CDS in the 2036-dataset. **(d)** Percentage of genes per numbers of CDS in the 723-dataset. **(e)** Percentage of gene families per percentage of genes having a single CDS in the 723-dataset.

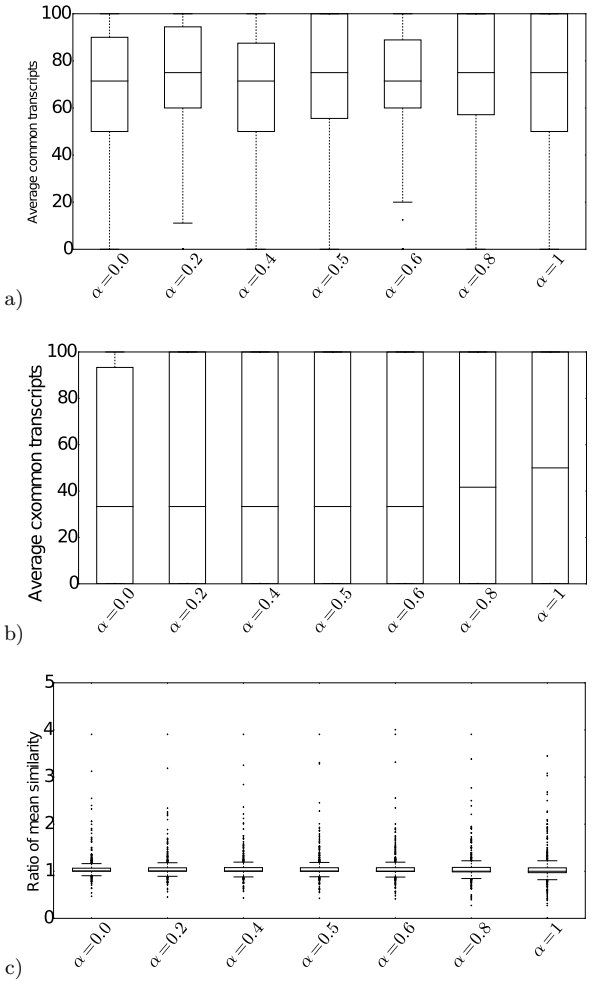


Fig. 6: **Common representative transcripts**: Boxplot of percentages of common representative transcripts in gene families between the SHT and the LT methods per values of α **(a)** in the 723-dataset, **(b)** when considering only genes having more than one transcript in the 723-dataset. **(c)** Boxplot of ratios of the mean similarity between the SHT transcripts over the mean similarity between the LT transcripts.

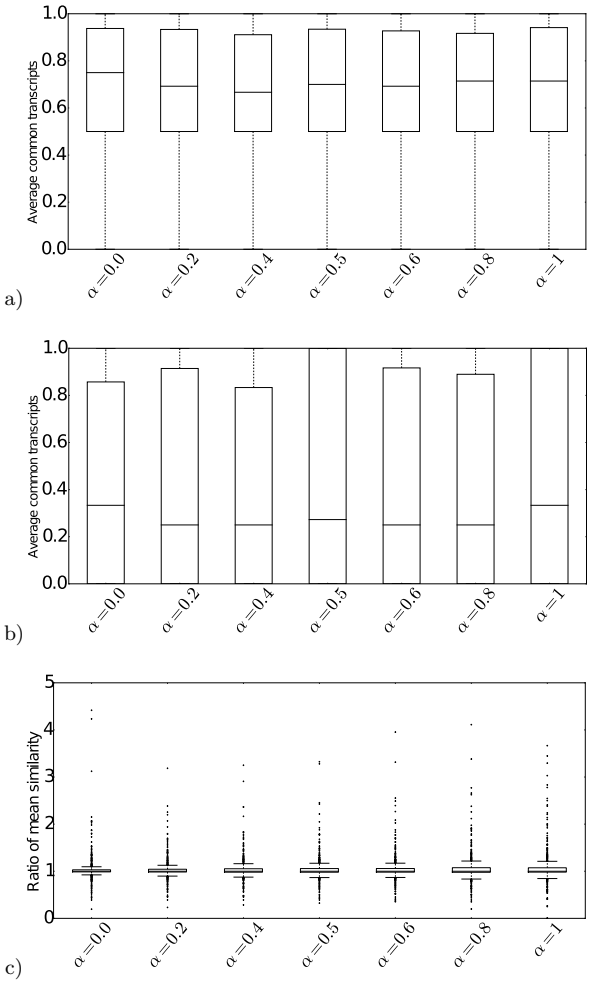


Fig. 7: **Common representative transcripts**: Boxplot of percentages of common representative transcripts in gene families between the SHT and the IsoSel methods per values of α **(a)** in the 723-dataset, **(b)** when considering only genes having more than one transcript in the 723-dataset. **(c)** Boxplot of ratios of the mean similarity between the SHT transcripts over the mean similarity between the IsoSel transcripts.

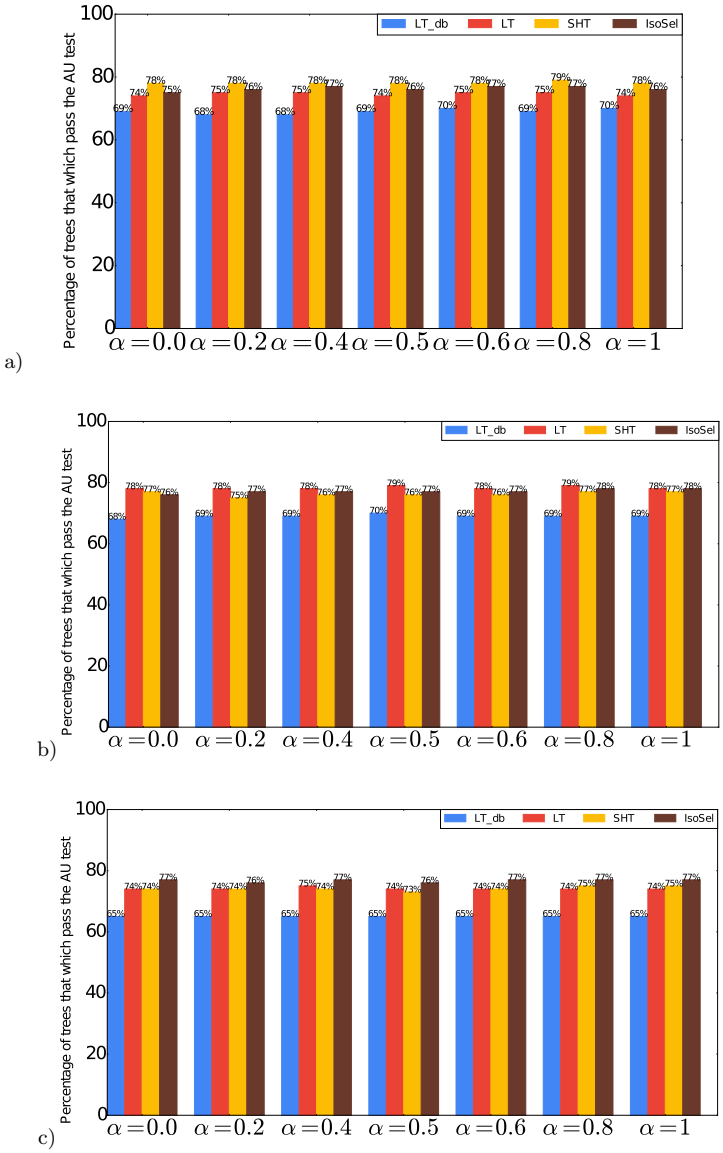


Fig. 8: **AU test**: Percentage of trees from each method (LT_db, LT, SHT, IsoSel) which pass the AU test per values of α. **(a)** When the transcript sequences selected by SHT are used as gene sequences for AU test. **(b)** Same comparison when transcript sequences selected by LT are used. **(c)** Same comparison when transcript sequences selected by IsoSel are used.

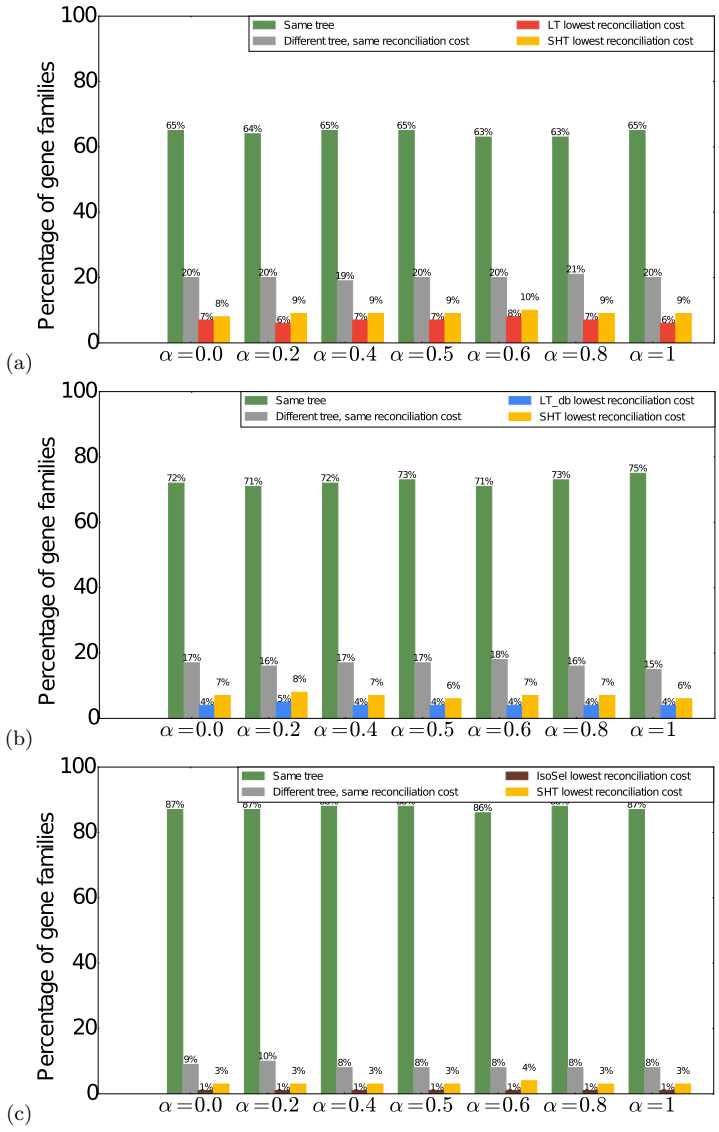


Fig. 9: **Reconciliation cost**: **(a)** Percentage of gene families for which the SHT and the LT gene trees are identical (green), for which the SHT and the LT gene trees differ but that they have the same reconciliation cost with the species tree (grey), for which the LT gene tree has a lower reconciliation cost (red), and for which the SHT gene tree has a lower reconciliation cost (yellow), per values of α. **(b)** Same for the comparison between the SHT and the LT_db gene trees. **(c)** Same for the comparison between the SHT and the IsoSel gene trees.

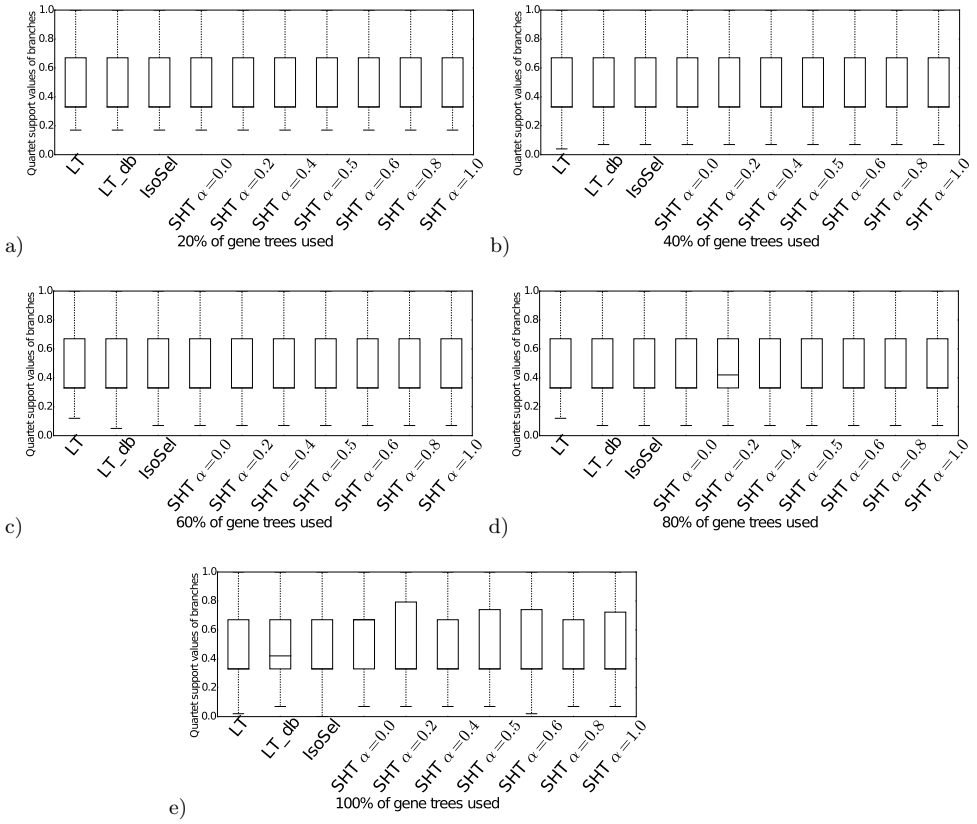


Fig. 10: **Species trees:** Quartet support values of branches for species tree deduced from the set of gene trees build by each of approach LT_db, LT, IsoSel, SHT. **(a)** When 20% of gene trees are used as input of Astral to build species tree. **(b)** Same when 40% of gene are used. **(c)** Same when 60% of gene are used. **(d)** Same when 80% of gene are used. **(e)** Same when 100% of gene are used.

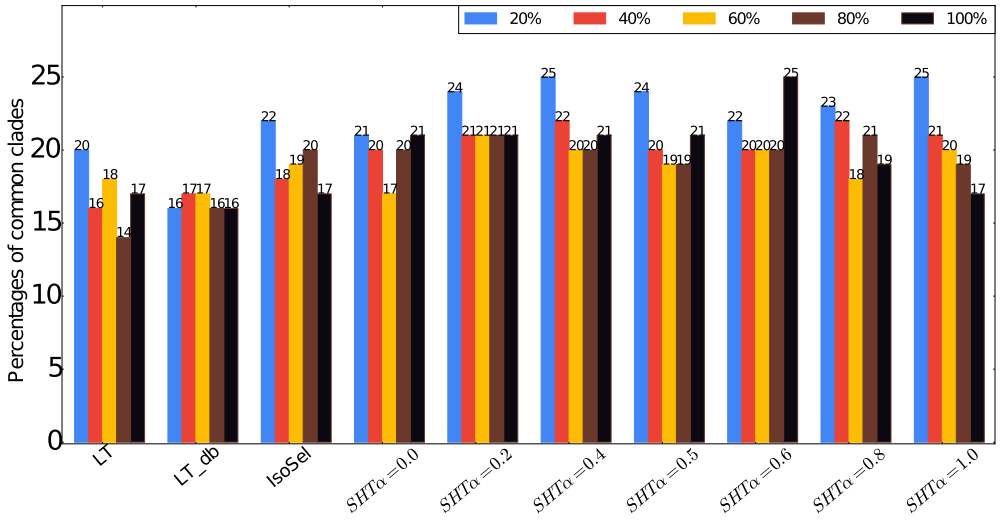


Fig. 11: Percentages of common clades between each reconstruct species trees and the initial Ensembl species tree when varying the percentage of used gene trees from.

Chapitre 6

Outils de visualisation de réconciliation à trois échelles : transcrits, gènes, espèces

6.1 Introduction

La phylogénie est l'étude de l'évolution ou de la coévolution d'un ensemble de taxons ou d'organismes. L'étude de l'évolution permet de comprendre comment un ensemble de gènes, d'espèces ou de transcrits a évolué, et la coévolution, quant à elle, représente l'évolution conjointe de deux taxons à deux échelles superposées, par exemple parasites et hôtes, gènes et espèces, transcrits et gènes. Les études portant sur l'évolution et la coévolution requièrent des outils permettant la visualisation, l'édition d'arbres phylogénétiques et la réconciliation entre arbres à différentes échelles. De plus, ces outils sont également utiles pour illustrer les phylogénies développées à l'aide des outils de reconstruction. De nombreux outils ont été développés pour remplir ces fonctionnalités et chacun a des propriétés spécifiques. Ce chapitre est structuré comme suit : la section 6.2 présente l'état de l'art des outils de visualisation d'arbres phylogénétiques. La section 6.3 présente les limites des outils existants. La section 6.4 présente DoubleRecViz, un outil de visualisation à trois échelles.

6.2 Outils de visualisation de phylogénies existants

Il existe deux grandes familles d'outils permettant la visualisation d'arbres phylogénétiques : ceux dédiés à la visualisation d'arbres phylogénétiques, et ceux dédiés à la visualisation de coévolutions.

6.2.1 Visualisation d'arbres phylogénétiques

Ces outils sont dédiés à la visualisation de l'évolution d'un ensemble de taxons qui peuvent être un ensemble d'espèces, de gènes, de transcrits ou de tout autre type de séquences reliées. Les principaux outils offrant ce type de visualisation sont décrits ci-dessous.

iTOL

iTOL[38] est un outil accessible via un navigateur qui permet la visualisation et la manipulation d'arbres phylogéniques. iTOL permet la visualisation d'arbres ayant jusqu'à 100 000 feuilles, et offre aux utilisateurs une inter-activité totale qui se base sur les technologies web récentes tels que HTML 5, JS et Canvas. iTOL permet à ses utilisateurs de créer un compte sur son site afin de sauvegarder leurs arbres. Les principales fonctionnalités de iTOL comptent l'édition d'arbres, les différents modes de visualisation disponibles, l'interaction avec les différents composants des arbres et l'exportation des arbres sous différents formats, dont des formats graphiques.

Dendroscope

Dendroscope[30] est un outil permettant la visualisation interactive d'arbres phylogénétiques et de graphiques. Les principales fonctionnalités de Dendroscope comportent la visualisation d'arbres de très grande taille, la recherche dans l'arbre, l'édition des feuilles, l'annotation de l'arbre et plusieurs modes de visualisation.

ETE3

ETE3[29] est une bibliothèque qui simplifie la reconstruction, l'analyse et la visualisation d'alignements de séquences multiples et d'arbres phylogénétiques. Les princi-

pales fonctionnalités de ETE3 comptent la reconstruction de phylogénies, la visualisation d'arbres phylogénétiques et le calcul de la réconciliation entre deux arbres. La visualisation de la réconciliation fournie par ETE3 permet d'identifier les événements de duplications et pertes de gènes dans un arbre.

FigTree

FigTree (http://tree.bio.ed.ac.uk/software/figtree/) est un programme de visualisation d'arbres phylogénétiques. Il possède une interface graphique qui permet aux utilisateurs d'éditer les propriétés des arborescences telles que les positions d'enracinement et les étiquettes de noeud. Les graphiques d'arbres peuvent être exportés au format PDF.

ggtree

ggtree[71] est un outil de visualisation d'arbres phylogénétiques qui permet l'annotation, la coloration, la rotation, la réduction et l'agrandissement des clades, et l'exportation des arbres sous divers formats.

MEGA

MEGA[27] est une bibliothèque qui permet la construction et la visualisation d'arbres phylogénétiques. La visualisation d'arbres avec MEGA offre divers modes de visualisation, d'enracinement d'arbres et l'exportation d'arbres au format PDF.

6.2.2 Visualisation d'une coévolution

Cette seconde catégorie d'outils permet de représenter l'évolution conjointe de deux ensembles de taxons à deux échelles différentes. Il peut s'agir par exemple de représenter l'évolution d'un ensemble de gènes dans un arbre d'espèces ou d'un ensemble de transcrits dans un arbre de gènes. Les principaux outils offrant ce type de visualisation sont décrits ci-dessous.

RecPhyloXML

RecPhyloXML[18] est un outil permettant de visualiser un arbre de gènes réconcilié avec un arbre d'espèces. RecphyloXML propose un nouveau format basé sur XML pour la représentation de la réconciliation entre un arbre de gènes et un arbre d'espèces. Cet outil permet la représentation des événements de duplication, de spéciation et de perte de gènes et l'exportation des graphiques d'arbres.

SylvX

SylvX[9] est un outil qui permet la représentation graphique d'arbres imbriqués. Il offre un ensemble d'opérations graphiques qui incluent le déplacement, la coloration et l'édition des composants de l'arbre. Il permet de visualiser des phylogénies comportant un grand nombre d'événements évolutifs. La réconciliation entre deux arbres est représentée sous le format newick. Il permet également l'exportation des graphiques d'arbres.

Primetv

Primetv[58] est un outil de visualisation de réconciliation de deux arbres. Il intègre des fonctionnalités d'édition, de coloration et de modification des clades. La réconciliation entre arbres est représentée au format newick. Il permet également l'exportation des graphiques d'arbres.

6.3 Limites des outils de visualisation de phylogénie

La principale limite des outils actuels de visualisation d'arbres vient du fait qu'ils proposent uniquement deux types de visualisation, à savoir : d'une part la visualisation d'un arbre, d'autre part la visualisation de deux arbres réconciliés avec un arbre d'espèces. Ces outils ne permettent pas de visualiser conjointement trois échelles d'évolution, telles que transcrits-gènes-espèces. Nous avons développé l'outil DoubleRecViz pour répondre à ce besoin spécifique.

6.4 Article : «DoubleRecViz : A Web-Based Tool for Visualizing Transcript-Gene-Species reconciliation»

Afin de pallier aux limites des outils actuels de visualisation de réconciliation phylogénétique, à savoir la non prise en charge de la visualisation de plus de deux arbres reconcicliés, nous avons développé DoubleRecViz qui est un outil permettant de visualiser des arbres phylogénétiques réconciliés à trois niveaux d'évolution : transcrits, gènes, espèces. Il permet de visualiser simultanément l'évolution d'un ensemble de transcrits dans un arbre de gènes, l'évolution d'un ensemble de gènes dans un arbre d'espèces. La visualisation est interactive et permet à l'utilisateur d'éditer les arbres réconciliés.

J'ai conçu l'étude avec Pr. Ouangraoua. J'ai conçu l'algorithme et développé le programme. J'ai rédigé la documentation avec Pr. Ouangraoua. Tahiri et Parmer m'ont guidé pour l'utilisation de la bibliothèque Dash. Qi a développé le serveur web. J'ai rédigé le manuscrit avec Pr. Ouangraoua. Tous les auteurs ont lu et approuvé le manuscrit final. L'article à été soumis au journal Bioinformatics.

Bioinformatics
doi.10.1093/bioinformatics/xxxxxx
Advance Access Publication Date: Day Month Year
Manuscript Category

OXFORD

Phylogenetics

DoubleRecViz: A Web-Based Tool for Visualizing Transcript-Gene-Species reconciliation

Esaie Kuitche [1,*], Yanchun Qi [1,*], Nadia Tahiri [2], Jack Parmer [2] and Aïda Ouangraoua [1]

[1]Department of Computer Science, University of Sherbrooke, Sherbrooke, J1K2R1, Canada,
[2]Plotly Inc, 5555 Avenue de Gaspe, Montreal, Quebec, H2T2A3, Canada

*To whom correspondence should be addressed.

Associate Editor: XXXXXXX

Received on XXXXX; revised on XXXXX; accepted on XXXXX

Abstract

Motivation: A phylogenetic tree reconciliation is a mapping of two phylogenetic trees which represents the co-evolution of two sets of taxa (eg. parasite-host co-evolution, gene-species co-evolution). The reconciliation framework was extended to allow modeling the co-evolution of three sets of taxa such as transcript-gene-species co-evolutions. Several web-based tools have been developed for the display and manipulation of phylogenetic trees and co-phylogenetic trees involving two trees, but there currently exists no tool for visualizing the joint reconciliation between three phylogenetic trees.
Results: Here, we present DoubleRecViz, a web-based tool for visualizing and editing double reconciliations between phylogenetic trees at three levels: transcript, gene and species. DoubleRecViz extends the RecPhyloXML model –developed for gene-species tree reconciliation– to represent joint transcript-gene and gene-species tree reconciliations. It is implemented using the Dash library, which is a toolbox that provides dynamic visualization functionalities for web data visualization in Python.
Availability and implementation: DoubleRecViz is available through a web server at https://doublerecviz.cobius.usherbrooke.ca. The source code and information about installation procedures are also available at https://github.com/UdeS-CoBIUS/DoubleRecViz.
Contact: Esaie.Kuitche.Kamela@USherbrooke.ca;Yanchun.Qi@USherbrooke.ca
Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

Phylogenetic tree visualization is important for comparative and evolutionary studies. Numerous phylogenetic tree viewers have been developed for the display, manipulation and exploration of phylogenetic trees (eg. Evolview [4], ETE3 [6], iTOL [8], GGtree [11]). Phylogenetic tree reconciliation is a common approach in evolutionary biology used to build co-phylogenetic trees, which are pairs of phylogenetic trees with a mapping among their leaves. Co-phylogenetic trees represent the co-evolution of pairs of sets of taxa such as parasites and hosts, or genes and species. The visualization of a co-phylogenetic tree allows to annotate the nodes of the tree with the underlying co-evolutionary events. Several models and tools have been developed for the representation and the visualization of co-phylogenetic trees (eg. [1, 5], RecPhyloVisu [3], SylvX [2], RecPhyloXML [3], Primetv [10]). The concept of reconciliation was recently extended to jointly model the multiscale mapping between three phylogenetic trees. For instance, the Domain-Gene-Species tree reconciliation maps a domain tree onto a gene tree which is mapped onto a species tree [9]. It allows to account for the proteic domains that compose genes in order to reconstruct detailed histories of domain and gene family evolution in eukaryote species. Similarly, the Trancript-Gene-Species tree reconciliation maps a transcript tree onto a gene tree which is mapped onto a species tree [7]. This framework allows to account for alternative splicing and to reconstruct the detailed histories of sets of alternative transcripts and gene families of eukaryote species. A visualisation tool for the display and the manipulation of three-scale co-phylogenies is currently not available. To this end, we adopted and enriched the recPhyloXML format [3] –which is an extension of the phyloXML format for the representation of gene-species reconciliation– by adding new tags for the representation of transcript-gene reconciliations. We present DoubleRecViz, a transcript-gene-species reconciliation tree viewer based

on the extended recPhyloXML format and using the Dash Python library (https://plot.ly/dash/) which allows dynamic visualization of web data.

2 Materials and methods

The recPhyloXML and recGeneTreeXML grammars are extensions of the standard PhyloXML format which were designed to describe gene-species tree reconciliation [3]. They were created in order to allow interoperability between the various scripts and softwares used in gene-species co-phylogeny studies. In this format, a gene-species tree reconciliation (<recPhylo> tag) contains a species tree (<spTree> tag) and one or more reconciled gene trees (<recGeneTree> tag). The recGeneTreeXML grammar has enriched the PhyloXML vocabulary by adding the <eventsRec> tag inside a <clade> tag to allow the representation of the sequence of evolutionary events associated to branches and nodes of a gene tree, including the position of these events on the associated species tree. The branches and nodes of the gene tree can be annotated with evolutionary events such as speciation (<speciation> tag for bifurcation nodes), gene duplication (<duplication> tag for bifurcation nodes), gene loss (<loss> tag for leaf nodes), horizontal gene transfer (<branchingOut> for bifurcation nodes to specify the origin of a transfer, and <transferBack> tag for branches to specify the destination of the transfer), and end of lineage (<leaf> tag for leaf nodes). We have extended the recPhyloXML format by adding the doubleRecPhyloXML and the recTransTreeXML grammars to describe transcript-gene-species and transcript-gene tree reconciliations. In addition to the types of evolutionary events inherited from the recGeneTreeXML grammar and corresponding to bifurcation or leaf nodes in reconciled gene trees (speciation, duplication, branchingOut, leaf), the nodes of a reconciled transcript tree can be annotated with the following evolutionary events: • a transcript creation (<creation> tag) which describes a transcript lineage undergoing a bifurcation due to the creation of a new lineage of transcript isoforms ; • a transcript loss (<loss> tag) which describes the loss of a transcript isoform. DoubleRecViz is written in Python, and makes use of the Dash Python library (https://plot.ly/dash/) –which provides dynamic visualization functionalities for scientific web data– to display the transcript-gene-species reconciled trees.

3 Features

DoubleRecViz takes as input an XML file which represents a <doubleRecPhylo> object containing a species tree (<spTree> object) followed by one or more sets, each composed of a reconciled gene tree (<recGeneTree> object) followed by zero or more reconciled transcript trees (<recTranscriptTree> object). A detailed description of the doubleRecPhyloXML format is available at https://doublerecviz.cobius.usherbrooke.ca/. DoubleRecViz generates a graphical representation of each gene-species tree reconciliations followed by the corresponding transcript-gene tree reconciliations. Evolutionary events in the reconciled gene trees are displayed as in RecPhyloVisu [3] (i.e gene duplication as square, gene loss as cross mark, gene transfer origin as diamond). Evolutionary events in the reconciled transcript tree are displayed as follows: transcript creation as triangle, and transcript creation as cross mark. For each reconciliation tree, the reconciliation cost is computed and displayed. The reconciliation cost for a gene-species reconciliation is the number of gene duplication, loss and transfer (branching out) nodes in the reconciled gene tree, and the reconciliation cost for a transcript-gene reconciliation is the number of transcript creation and loss nodes in the reconciled transcript tree. The display generated by DoubleRecViz can be exported as a figure file for further manipulation. Figure 1 shows an example of display obtained for a reconciliation between a species tree, a gene tree and a transcript tree. The corresponding <doubleRecPhylo> object can be seen in Supplementary Figure S1. This program is provided with sample data inputs of different sizes.

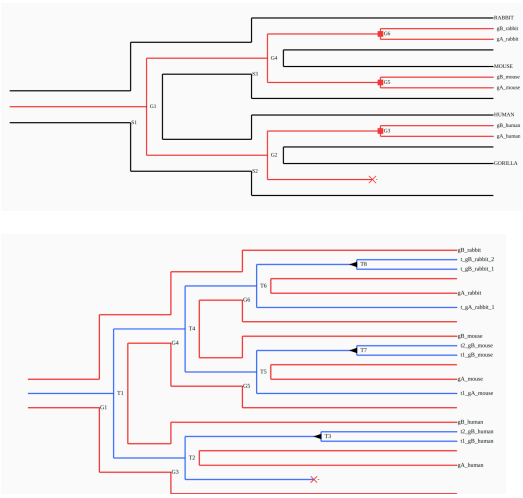


Fig. 1. Visualization of the transcript-gene-species reconciliation represented by the <doubleRecPhylo> object shown in Supplementary Figure S1. The top graphic illustrates the gene-species reconciliation with the species tree in black color and the gene tree in red color. The bottom graphic illustrates the transcript-gene reconciliation with the gene tree in red color and the transcript tree in blue color.

4 Conclusion

DoubleRecViz is the first automated tool for the visualization and the exploration of transcript-gene-species tree reconciliations. Thanks to an extension of the recPhyloXML format [3] and the use of the Dash Python library (https://plot.ly/dash/), DoubleRecViz allows to efficiently visualize double reconciliations between transcript, gene and species trees.

Acknowledgements

This work was supported by the Mitacs accelaration program and Plotly Inc (Grant IT11886), the Canada Research Chair (CRC Tier 2 Grant 950-230577), and the Faculty of Science of Université de Sherbrooke.

References

[1]Tiziana Calamoneri, Valentino Di Donato, Diego Mariottini, and Maurizio Patrignani. Visualizing co-phylogenetic reconciliations. *Theoretical Computer Science*, 2020.

[2]François Chevenet, Jean-Philippe Doyon, Celine Scornavacca, Edwin Jacox, Emmanuelle Jousselin, and Vincent Berry. Sylvx: a viewer for phylogenetic tree reconciliations. *Bioinformatics*, 32(4):608–610, 2015.

[3]Wandrille Duchemin, Guillaume Gence, Anne-Muriel Arigon Chifolleau, Lars Arvestad, Mukul S Bansal, Vincent Berry, Bastien Boussau, François Chevenet, Nicolas Comte, Adrián A Davín, et al. Recphyloxml: a format for reconciled gene trees. *Bioinformatics*, 34(21):3646–3652, 2018.

[4]Zilong He, Huangkai Zhang, Shenghan Gao, Martin J Lercher, Wei-Hua Chen, and Songnian Hu. Evolview v2: an online visualization and management tool for customized and annotated phylogenetic trees. *Nucleic acids research*, 44(W1):W236–W241, 2016.

[5]Katharina T Huber, Vincent Moulton, Marie-France Sagot, and Blerina Sinaimeri. Exploring and visualizing spaces of tree reconciliations. *Systematic biology*, 68(4):607–618, 2019.

[6]Jaime Huerta-Cepas, François Serra, and Peer Bork. Ete 3: reconstruction, analysis, and visualization of phylogenomic data. *Molecular biology and evolution*, 33(6):1635–1638, 2016.

[7]Esaie Kuitche, Manuel Lafond, and Aïda Ouangraoua. Reconstructing protein and gene phylogenies using reconciliation and soft-clustering. *Journal of bioinformatics and computational biology*, 15(06):1740007, 2017.

[8]Ivica Letunic and Peer Bork. Interactive tree of life (itol) v4: recent updates and new developments. *Nucleic acids research*, 47(W1):W256–W259, 2019.

[9]Lei Li and Mukul S Bansal. An integrated reconciliation framework for domain, gene, and species level evolution. *IEEE/ACM transactions on computational biology and bioinformatics*, 16(1):63–76, 2018.

[10]Bengt Sennblad, Eva Schreil, Ann-Charlotte Berglund Sonnhammer, Jens Lagergren, and Lars Arvestad. Primetv: a viewer for reconciled trees. *BMC bioinformatics*, 8(1):148, 2007.

[11]Guangchuang Yu, David K Smith, Huachen Zhu, Yi Guan, and Tommy Tsan-Yuk Lam. ggtree: an r package for visualization and annotation of phylogenetic trees with their covariates and other associated data. *Methods in Ecology and Evolution*, 8(1):28–36, 2017.

Conclusion

L'objectif de cette thèse est de développer des modèles et des méthodes pour la segmentation de séquences et la reconstruction de phylogénies de transcrits et de gènes tenant compte de l'épissage alternatif des transcrits. L'épissage alternatif permet à un gène de produire plusieurs transcrits ayant parfois des fonctions différentes et permet ainsi aux gènes de se diversifier fonctionnellement.

Pour atteindre l'objectif de cette thèse, nous apportons quatre contributions théoriques et pratiques majeures.

Notre première contribution est la démonstration de la nécessité de distinguer les arbres de gènes des arbres de transcrits. Nous proposons des modèles d'évolution de transcrits dans des arbres de gènes. Nous présentons le concept de double réconciliation qui consiste à comparer simultanément un arbre de transcrits, un arbre de gènes et un arbre d'espèces. Nous proposons une méthode de segmentation avec chevauchement des séquences de transcrits selon leurs similarités. Nous posons deux problèmes algorithmiques de reconstruction d'arbres de gènes et de transcrits selon les modèles d'évolution présentés. Une application des algorithmes proposés sur des données réelles permet de démontrer la pertinence de nos modèles.

Notre seconde contribution est un outil de simulation de l'évolution des séquences biologiques tenant compte de l'épissage alternatif. Nous avons décrit un modèle d'évolution conjoint de transcrits et de gènes. D'un point de vue théorique, le modèle proposé décrit comment un gène et tous ses composants évoluent, et comment les transcrits sont créés et évoluent au fil du temps. D'un point de vue pratique, nous fournissons une implémentation de ce modèle qui permet à un utilisateur de simuler l'évolution d'un ensemble de données, dont des événements d'épissage alternatif. Nous démontrons l'importance de ce travail en l'utilisant pour générer des données de réfé-

rences permettant d'évaluer des outils d'alignement et de segmentation de séquences biologiques.

Notre troisième contribution est la reconstruction des arbres de gènes tenant compte de la similarité en termes de structures d'épissage. Nous proposons une mesure de similarité entre transcrits qui tient compte de la structure exon-intron des transcrits, de la phase de traduction des transcrits et des événements d'épissage alternatif subis par chaque exon. Nous avons développé un algorithme de segmentation flou pour l'identification de groupe de transcrits orthologues. Pour y parvenir, nous avons utilisé une représentation latente des séquences afin de tirer avantage des propriétés géométriques des vecteurs pour mieux regrouper les séquences.

Notre quatrième contribution est un outil dédié à la visualisation de coévolution d'espèces, de gènes et de transcrits. La spécificité de l'outil réside dans la possibilité de visualiser simultanément trois échelles d'évolution. C'est-à-dire, la visualisation d'un arbre de transcrits imbriqué dans un arbre de gènes et la visualisation d'un arbre de gènes imbriqué dans un arbre d'espèces. De plus, l'outil de visualisation interactif permet de calculer le coût de double réconciliation entre trois échelles. Ce coût est la somme du coût de réconciliation entre un arbre de transcrits et un arbre de gènes, plus le coût de réconciliation entre l'arbre de gènes et un arbre d'espèces.

Les contributions de cette thèse s'inscrivent parmi les efforts visant à reconstruire avec précision les phylogénies de transcrits, de gènes, ainsi que leurs co-phylogénies. L'originalité de ces contributions découle du fait que nous sommes les premiers à considérer l'épissage alternatif dans la construction des arbres de gènes et la reconstruction de co-phylogénies à trois échelles : transcrits, gènes et espèces. Nous démontrons la nécessité de distinguer les arbres de protéines (transcrits de référence) des arbres de gènes, et de réconcilier trois niveaux d'évolution pour reconstruire l'histoire évolutive détaillée d'ensembles de transcrits et de familles de gènes dans des espèces eucaryotes. La prise en compte de l'épissage alternatif dans la reconstruction de phylogénies et l'ensemble des contributions de cette thèse ouvre la voie à de nouvelles perspectives pour une meilleure compréhension de l'évolution de l'épissage alternatif et la diversification fonctionnelle des gènes à travers diverses espèces.

Bibliographie

[1] Adel Ait-hamlat, Lelia Polit, Hugues Richard et Elodie Laine.
« Transcripts Evolutionary Conservation and Structural Dynamics give Insights into the Role of Alternative Splicing for the JNK Family ».
bioRxiv, page 119891, 2017.

[2] Cécile Ané.
« Detecting Phylogenetic Breakpoints and Discordance from Genome-wide Alignments for Species Tree Reconstruction ».
Genome Biology and Evolution, 3:246–258, 2011.

[3] B Edwin Blaisdell.
« A Measure of the Similarity of Sets of Sequences Not Requiring Sequence Alignment ».
Proceedings of the National Academy of Sciences, 83(14):5155–5159, 1986.

[4] Samuel Blanquart, Jean-Stéphane Varré, Paul Guertin, Amandine Perrin, Anne Bergeron et Krister M Swenson.
« Assisted Transcriptome Reconstruction and Splicing Orthology ».
BMC Genomics, 17(10):786, 2016.

[5] Brigitte Boeckmann, Marc Robinson-Rechavi, Ioannis Xenarios et Christophe Dessimoz.
« Conceptual Framework and Pilot Study to Benchmark Phylogenomic Databases Based on Reference Gene Trees ».
Briefings in Bioinformatics, 12(5):423–435, 2011.

[6] Volker Brendel, Liqun Xing et Wei Zhu.
« Gene Structure Prediction from Consensus Spliced Alignment of Multiple ESTs

Matching the Same Genomic Locus ».
Bioinformatics, 20(7):1157–1169, 2004.

[7] Reed A Cartwright.
« DNA Assembly with Gaps (Dawg) : Simulating Sequence Evolution ».
Bioinformatics, 21(Suppl_3):iii31–iii38, 2005.

[8] Kevin Chen, Dannie Durand et Martin Farach-Colton.
« Notung : Dating Gene Duplications using Gene Family Trees ».
Dans *Proceedings of the Fourth Annual International Conference on Computational Molecular Biology*, pages 96–106. ACM, 2000.

[9] François Chevenet, Jean-Philippe Doyon, Celine Scornavacca, Edwin Jacox, Emmanuelle Jousselin et Vincent Berry.
« SylvX : a Viewer for Phylogenetic Tree Reconciliations ».
Bioinformatics, 32(4):608–610, 2015.

[10] Yann Christinat et Bernard ME Moret.
« Inferring Transcript Phylogenies ».
BMC Bioinformatics, 13(9):S1, 2012.

[11] Yann Christinat et Bernard ME Moret.
« A Transcript Perspective on Evolution ».
IEEE/ACM Transactions on Computational Biology and Bioinformatics, 10(6):1403–1411, 2013.

[12] Yujin Chung et Cécile Ané.
« Comparing two Bayesian Methods for Gene Tree/Species Tree Reconstruction : Simulations with Incomplete Lineage Sorting and Horizontal Gene Transfer ».
Systematic Biology, 60(3):261–275, 2011.

[13] Aaron CE Darling, Bob Mau, Frederick R Blattner et Nicole T Perna.
« Mauve : Multiple Alignment of Conserved Genomic Sequence with Rearrangements ».
Genome Research, 14(7):1394–1403, 2004.

[14] Ruchira S Datta, Christopher Meacham, Bushra Samad, Christoph Neyer et Kimmen Sjölander.
« Berkeley PHOG : PhyloFacts Orthology Group Prediction Beb Server ».
Nucleic Acids Research, 37(suppl_2):W84–W89, 2009.

[15] William HE Day, David S Johnson et David Sankoff.
« The Computational Complexity of Inferring Rooted Phylogenies by Parsimony ».
Mathematical Biosciences, 81(1):33–42, 1986.

[16] I Dondoshansky et Y Wolf.
« Blastclust (NCBI Software Development Toolkit) ».
NCBI, Bethesda, Md, 14, 2002.

[17] Wandrille Duchemin, Yoann Anselmetti, Murray Patterson, Yann Ponty, Sèverine Bérard, Cedric Chauve, Celine Scornavacca, Vincent Daubin et Eric Tannier.
« DeCoSTAR : Reconstructing the Ancestral Organization of Genes or Genomes using Reconciled Phylogenies ».
Genome Biology and Evolution, 9(5):1312–1319, 2017.

[18] Wandrille Duchemin, Guillaume Gence, Anne-Muriel Arigon Chifolleau, Lars Arvestad, Mukul S Bansal, Vincent Berry, Bastien Boussau, François Chevenet, Nicolas Comte, Adrián A Davín et others.
« RecPhyloXML : a format for reconciled gene trees ».
Bioinformatics, 34(21):3646–3652, 2018.

[19] David M Emms et Steven Kelly.
« OrthoFinder : Solving Fundamental Biases in Whole Genome Comparisons Dramatically Improves Orthogroup Inference Accuracy ».
Genome Biology, 16(1):157, 2015.

[20] Walter M Fitch.
« Toward Finding the Tree of Maximum Tarsimony ».
Dans *Proceedings of the 8th International Conference on Numerical Taxonomy*, pages 189–230. Freeman, 1975.

[21] William Fletcher et Ziheng Yang.
« INDELible : a Flexible Simulator of Biological Sequence Evolution ».
Molecular Biology and Evolution, 26(8):1879–1888, 2009.

[22] Paweł Górecki, Oliver Eulenstein et Jerzy Tiuryn.
« Evolutionary cCosts in Gene-Species Reconciliation ».

[23] Paweł Górecki et Jerzy Tiuryn.
« Inferring Evolutionary Scenarios in the Duplication, Loss and Horizontal Gene Transfer Model ».
Dans *Logic and Program Semantics*, pages 83–105. Springer, 2012.

[24] Dan Gusfield.
Algorithms on Strings, Trees, and Sequences : Computer Science and Computational Biology.
Cambridge University Press, 1997.

[25] Matthew W Hahn.
« Bias in Phylogenetic Tree Reconciliation Methods : Implications for Vertebrate Genome Evolution ».
Genome Biology, 8(7):R141, 2007.

[26] Barry G Hall.
« Simulating DNA coding sequence evolution with EvolveAGene 3 ».
Molecular Biology and Evolution, 25(4):688–695, 2008.

[27] Barry G Hall.
« Building Phylogenetic Trees from Molecular Data with MEGA ».
Molecular Biology and Evolution, 30(5):1229–1235, 2013.

[28] Jaime Huerta-Cepas, Salvador Capella-Gutierrez, Leszek P Pryszcz, Ivan Denisov, Diego Kormes, Marina Marcet-Houben et Toni Gabaldon.
« PhylomeDB v3. 0 : an Expanding Repository of Genome-wide Collections of Trees, Alignments and Phylogeny-based Orthology and Paralogy Predictions ».
Nucleic Acids Research, 39(suppl_1):D556–D560, 2010.

[29] Jaime Huerta-Cepas, François Serra et Peer Bork.
« ETE 3 : Reconstruction, Analysis, and Visualization of Phylogenomic Data ».
Molecular Biology and Evolution, 33(6):1635–1638, 2016.

[30] Daniel H Huson, Daniel C Richter, Christian Rausch, Tobias Dezulian, Markus Franz et Regula Rupp.
« Dendroscope : An Interactive Viewer for Large Phylogenetic Trees ».
BMC Bioinformatics, 8(1):460, 2007.

[31] Safa Jammali, Jean-David Aguilar, Esaie Kuitche et Aïda Ouangraoua.
« SplicedFamAlign : CDS-to-gene Spliced Alignment and Identification of Trans-

cript Orthology Groups ».
BMC Bioinformatics, 20(3):133, 2019.

[32] Mark Johnson, Irena Zaretskaya, Yan Raytselis, Yuri Merezhuk, Scott McGinnis et Thomas L Madden.
« NCBI BLAST : a Better Web Interface ».
Nucleic Acids Research, 36(suppl_2):W5–W9, 2008.

[33] Yuri Kapustin, Alexander Souvorov, Tatiana Tatusova et David Lipman.
« Splign : Algorithms for Computing Spliced Alignments with Identification of Paralogs ».
Biology Direct, 3(1):20, 2008.

[34] Abdellali Kelil, Shengrui Wang et Ryszard Brzezinski.
« CLUSS2 : an Alignment-Independent Algorithm for Clustering Protein Families with Multiple Biological Functions ».
International Journal of Computational Biology and Drug Design, 1(2):122–140, 2008.

[35] Esaie Kuitche, Manuel Lafond et Aïda Ouangraoua.
« Reconstructing Protein and Gene Phylogenies using Reconciliation and Soft-clustering ».
Journal of Bioinformatics and Computational Biology, 15(06):1740007, 2017.

[36] Manuel Lafond, Cedric Chauve, Nadia El-Mabrouk et Aida Ouangraoua.
« Gene Tree Construction and Correction Using Supertree and Reconciliation ».
IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB), 15(5):1560–1570, 2018.

[37] Marcus Lechner, Sven Findeiss, Lydia Steiner, Manja Marz, Peter F Stadler et Sonja J Prohaska.
« Proteinortho : Detection of (co-) Orthologs in Large-Scale Analysis ».
BMC Bioinformatics, 12(1):124, 2011.

[38] Ivica Letunic et Peer Bork.
« Interactive Tree of Life (iTOL) v3 : an Online Tool for the Display and Annotation of Phylogenetic and other Trees ».
Nucleic Acids Research, 44(W1):W242–W245, 2016.

[39] Vladimir I Levenshtein.
« Binary Codes Capable of Correcting Deletions, Insertions, and Reversals ».
Dans *Soviet Physics Doklady*, volume 10, pages 707–710, 1966.

[40] Lei Li et Mukul S Bansal.
« An Integer Linear Programming Solution for the Domain-Gene-Species Reconciliation Problem ».
Dans *Proceedings of the 2018 ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics*, pages 386–397. ACM, 2018.

[41] Lei Li et Mukul S Bansal.
« An Integrated Reconciliation Framework for Domain, Gene, and Species Level Evolution ».
IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB), 16(1):63–76, 2019.

[42] Li Li, Christian J Stoeckert et David S Roos.
« OrthoMCL : Identification of Ortholog Groups for Eukaryotic Genomes ».
Genome Research, 13(9):2178–2189, 2003.

[43] Fred R McMorris, David B Meronk et Dean A Neumann.
« A View of some Consensus Methods for Trees ».
Dans *Numerical Taxonomy*, pages 122–126. Springer, 1983.

[44] Huaiyu Mi, Anushya Muruganujan et Paul D Thomas.
« PANTHER in 2013 : Modeling the Evolution of Gene Function, and other Gene Attributes, in the Context of Phylogenetic Trees ».
Nucleic Acids Research, 41(D1):D377–D386, 2012.

[45] Sayyed Auwn Muhammad.
« *Probabilistic Modelling of Domain and Gene Evolution* ».
Thèse de doctorat, KTH Royal Institute of Technology, 2016.

[46] Saul B Needleman et Christian D Wunsch.
« A General Method Applicable to the Search for Similarities in the Amino Acid Sequence of two Proteins ».
Journal of Molecular Biology, 48(3):443–453, 1970.

[47] Emmanuel Noutahi et Nadia El-Mabrouk.
« GATC : a Genetic Algorithm for Gene Tree Construction under the

Duplication-Transfer-Loss Model of Evolution ».
BMC genomics, 19(2):102, 2018.

[48] Javier Oyarzun.
« Inférence Bayésienne pour la Reconstruction d'Arbres Phylogénétiques », 2006.

[49] Andy Pang, Andrew D Smith, Paulo AS Nuin et Elisabeth RM Tillier.
« SIMPROT : Using an Empirically Determined Indel Distribution in Simulations of Protein Evolution ».
BMC Bioinformatics, 6(1):236, 2005.

[50] Simon Penel, Anne-Muriel Arigon, Jean-François Dufayard, Anne-Sophie Sertier, Vincent Daubin, Laurent Duret, Manolo Gouy et Guy Perrière.
« Databases of Homologous Gene Families for Comparative Genomics ».
Dans *BMC Bioinformatics*, volume 10, page S3. BioMed Central, 2009.

[51] Héloïse Philippon, Alexia Souvane, Céline Brochier-Armanet et Guy Perrière.
« IsoSel : Protein Isoform Selector for phylogenetic reconstructions ».
PloS one, 12(3):e0174250, 2017.

[52] Peter Pipenbacher, Alexander Schliep, Sebastian Schneckener, Alexander Schönhuth, Dietmar Schomburg et Rainer Schrader.
« ProClust : Improved Clustering of Protein Sequences with an Extended Graph-Based Approach ».
Bioinformatics, 18(suppl_2):S182–S191, 2002.

[53] Leszek P Pryszcz, Jaime Huerta-Cepas et Toni Gabaldon.
« MetaPhOrs : Orthology and Paralogy Predictions from Multiple Phylogenetic Evidence using a Consistency-Based Confidence Score ».
Nucleic Acids Research, 39(5):e32–e32, 2010.

[54] Ashok Rajaraman et Jian Ma.
« Reconstructing Ancestral Gene Orders with Duplications Guided by Synteny Level Genome Reconstruction ».
BMC bioinformatics, 17(14):414, 2016.

[55] Matthew D Rasmussen et Manolis Kellis.
« A Bayesian Approach for Fast and Accurate Gene Tree Reconstruction ».
Molecular Biology and Evolution, 28(1):273–290, 2010.

[56] Naruya Saitou et Masatoshi Nei.
« The neighbor-joining method : a new method for reconstructing phylogenetic trees. ».
Molecular biology and evolution, 4(4):406–425, 1987.

[57] Fabian Schreiber, Mateus Patricio, Matthieu Muffato, Miguel Pignatelli et Alex Bateman.
« TreeFam v9 : a new website, more species and orthology-on-the-fly ».
Nucleic acids research, 42(D1):D922–D925, 2014.

[58] Bengt Sennblad, Eva Schreil, Ann-Charlotte Berglund Sonnhammer, Jens Lagergren et Lars Arvestad.
« Primetv : a Viewer for Reconciled Trees ».
BMC Bioinformatics, 8(1):148, 2007.

[59] Hidetoshi Shimodaira et Masami Hasegawa.
« CONSEL : for assessing the confidence of phylogenetic tree selection ».
Bioinformatics, 17(12):1246–1247, 2001.

[60] Botond Sipos, Tim Massingham, Gregory E Jordan et Nick Goldman.
« PhyloSim-Monte Carlo Simulation of Sequence Evolution in the R Statistical Computing Environment ».
BMC Bioinformatics, 12(1):104, 2011.

[61] Temple F Smith, Michael S Waterman et others.
« Identification of Common Molecular Subsequences ».
Journal of Molecular Biology, 147(1):195–197, 1981.

[62] Michener Sokal.
« A statistical method for evaluating systematic relationships ».
University of Kansas Science Bulletin, 38, 1958.

[63] Jens Stoye, Dirk Evers et Folker Meyer.
« Rose : Generating Requence Families. ».
Bioinformatics (Oxford, England), 14(2):157–163, 1998.

[64] Cory L Strope, Kevin Abel, Stephen D Scott et Etsuko N Moriyama.
« Biological Sequence Simulation for Testing Complex Evolutionary Hypotheses : Indel-Seq-Gen version 2.0 ».
Molecular Biology and Evolution, 26(11):2581–2593, 2009.

[65] Gergely J Szöllősi, Eric Tannier, Vincent Daubin et Bastien Boussau.
« The Inference of Gene Trees with Species Trees ».
Systematic Biology, 64(1):e42–e62, 2014.

[66] Albert J Vilella, Jessica Severin, Abel Ureta-Vidal, Li Heng, Richard Durbin et Ewan Birney.
« EnsemblCompara GeneTrees : Complete, Duplication-Aware Phylogenetic Trees in Vertebrates ».
Genome Research, 19(2):327–335, 2009.

[67] Susana Vinga et Jonas Almeida.
« Alignment-Free Sequence Comparison—a Review ».
Bioinformatics, 19(4):513–523, 2003.

[68] Tobias Wittkop, Dorothea Emig, Sita Lange, Sven Rahmann, Mario Albrecht, John H Morris, Sebastian Böcker, Jens Stoye et Jan Baumbach.
« Partitioning Biological Data with Transitivity Clustering ».
Nature Methods, 7(6):419, 2010.

[69] Tobias Wittkop, Dorothea Emig, Anke Truss, Mario Albrecht, Sebastian Böcker et Jan Baumbach.
« Comprehensive Cluster Analysis with Transitivity Clustering ».
Nature Protocols, 6(3):285, 2011.

[70] Pablo Yarza, Michael Richter, Jörg Peplies, Jean Euzeby, Rudolf Amann, Karl-Heinz Schleifer, Wolfgang Ludwig, Frank Oliver Glöckner et Ramon Rosselló-Móra.
« The All-Species Living Tree Project : a 16S rRNA-based Phylogenetic Tree of all Sequenced Type Strains ».
Systematic and Applied Microbiology, 31(4):241–250, 2008.

[71] Guangchuang Yu, David K Smith, Huachen Zhu, Yi Guan et Tommy Tsan-Yuk Lam.
« ggtree : an R Package for Visualization and Annotation of Phylogenetic Trees with their Covariates and other Associated Data ».
Methods in Ecology and Evolution, 8(1):28–36, 2017.